CLASSIFICATION OF TARGET PRIMITIVES WITH SONAR USING TWO NON-PARAMETRIC DATA FUSION METHODS

A THESIS
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING
AND THE INSTITUTE OF ENGINEERING AND SCIENCES
OF BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

By
Birsel Ayrulu
July 1996

CLASSIFICATION OF TARGET PRIMITIVES WITH SONAR USING TWO NON-PARAMETRIC DATA FUSION METHODS

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND

ELECTRONICS ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCES

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

Birsel Ayrulu

July 1996

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assist. Prof. Dr. Billur Barshan(Supervisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Ömer Morgül

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assist. Prof. Dr. Orhan Arıkan

Approved for the Institute of Engineering and Sciences:

Prof. Dr. Mehmet Baray
Director of Institute of Engineering and Sciences

to Aykut

ABSTRACT

CLASSIFICATION OF TARGET PRIMITIVES WITH SONAR USING TWO NON-PARAMETRIC DATA FUSION METHODS

Birsel Ayrulu
M.S. in Electrical and Electronics Engineering
Supervisor: Assist. Prof. Dr. Billur Barshan
July 1996

In this study, physical models are used to model reflections from target primitives commonly encountered in a mobile robot's environment. These targets are differentiated by employing a multi-transducer pulse/echo system which relies on both amplitude and time-of-flight (TOF) data in the feature fusion process, allowing more robust differentiation. Target features are generated as being evidentially tied to degrees of belief which are subsequently fused for multiple logical sonars at different geographical sites. Feature data from multiple logical sensors are fused with Dempster-Shafer rule of combination to improve the performance of classification by reducing perception uncertainty. Using three sensing nodes, improvement in differentiation is 20% without false decision, however, at the cost of additional computation. Simulation results are verified by experiments with real sonar systems. As an alternative method, neural networks are used for incorporating learning of identifying parameter relations of target primitives. Amplitude and time-of-flight measurements of 31 sensor pairs are fused with these neural networks. Improvement in differentiation is 72% with 28% false decision at the cost of elapsed time until the network learns these patterns. These two approaches help to overcome the vulnerability of echo amplitude to noise and enable the modeling of non-parametric uncertainty.

Keywords : ultrasonic transducers, sonar, target classification, sensor-based robotics, multi-sensor data fusion and integration, non-parametric data fusion, evidential reasoning, belief functions, Dempster-Shafer rule of combination, logical sensing, artificial neural networks.

ÖZET

HEDEF İLKELLERİN SONAR KULLANILARAK PARAMETRİK OLMAYAN İKİ YÖNTEMLE SINIFLANDIRILMASI

Birsel Ayrulu
Elektrik ve Elektronik Mühendisliği Bölümü Yüksek Lisans
Tez yöneticisi: Yrd. Doç. Dr. Billur Barshan
Temmuz 1996

Bu çalışmada, hareketli robot uygulamalarında sıkça karşılaşılan hedef ilkellerinden kaynaklanan yansımaların modellenmesinde fiziksel modeller kullanıldı. Genlik ve uçuş zamanı verilerine dayanan çok dönüştürücülü dürtü/yankı sistemleri ile hedef ilkelleri tanınmaya çalışıldı. Birden çok mantıksal sonar için atanan inanç değerleri taşıyan hedef özellikleri sürekli olarak birleştirildi. Algı belirsizliklerini azaltıp sınıflandırma başarısını arttırmak için yapılan bu birleştirme işlemi için Dempster-Shafer birleşim kuralı kullanıldı. Üç algılayıcı çifti kullanılarak sınıflandırma işleminde fazladan yapılan işlemlere karşın %20 ye varan bir başarım artışı elde edildi. İkinci bir yöntem olarak, tanıtım parametre yakınlıklarının beraberce öğrenilmesi için yapay sinir ağları kullanıldı. Bu ağ yapısında devrenin bu hedef ilkellerini öğreninceye kadar geçen zaman ve 31 algılayıcı çiftine rağmen genlik ve uçuş zamanı ölçümleri kullanılarak sınıflandırma işleminde %72 ye varan bir başarı ve %28 yanlış karar elde edildi. Bu çalışmada kullanılan iki yöntemin her biri yankı genliğinin gürültüye karşı olan duyarlılığının üstesinden gelmeye yardımcı olduğu gibi parametrik olmayan belirsizliklerin modellenmesini de sağlamaktadır.

Anahtar Kelimeler : sesötesi duyucular, engel sınıflandırma, robotik algılama, parametrik olmayan veri tümleşimi, Dempster-Shafer birleşim kuralı, mantıksal algılama, yapay sinir ağları.

ACKNOWLEDGMENTS

I would like to thank everyone who contributed to this thesis. First of all, I would like to thank Dr. Billur Barshan for her supervision, guidance, suggestions and encouragement throughout the development of this thesis.

I would like to thank Dr. Aydan M. Erkmen, Dr. İsmet Erkmen and Dr. Ömer Morgül for their helpful suggestions.

Many thanks to *Aykut Erdem* for his help and encouragement.

Many thanks also to Gül Aydın, Nur Kurt, Fatma Çalışkan and Lütfiye Durak for their help.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION

There is no single sensor which perfectly detects, locates and identifies targets under all circumstances. Although some sensors are more accurate at locating and tracking objects, they do not provide identity information or vice versa. This points to the need for combining data from multiple sensors via data fusion techniques. The primary aim of the data fusion is to combine data from multiple sensors to perform inferences that may not be possible from a single sensor. Data fusion applications span a wide domain including automatic target recognition, mobile robot navigation, target tracking, aircraft navigation and military applications [1, 2]. In robotics applications, data fusion enables intelligent sensors to be incorporated into overall operation of a robot so that they can interact with and operate in an unstructured environment without the complete control of a human operator [3].

Qualitative benefits which are gained through multisensor data fusion include improved operational performance, extended spatial coverage, reduced ambiguity of inferences, improved detection, improved system reliability and enhanced spatial resolution. Quantitatively, the benefit of multisensor data fusion is the improvement of the accuracy of inferences such as estimation of location or identity declaration.

Data fusion can be accomplished by using geometrically, geographically or physically different sensors at different levels of representation such as signal-level, pixel-level, feature-level and symbol-level fusion. In this study, we use physically identical sonar sensors to combine information when they are located at geographically different sensing sites. Feature-level fusion will be used to provide a system performing an object recognition task with additional features

that can be used to increase its recognition capabilities.

Problems associated with multisensor data fusion center around the methods used for modeling the error or uncertainty in sensory information, data fusion process and operation of the overall system. There are many frameworks for multisensor data fusion. In this study two of them are used: The first one is the Dempster-Shafer evidence processing which is used in the data fusion at the level of evidence. Second one is artificial neural networks which provide a fairly well-established formalism with which to model the multisensor fusion process.

One mode of sensing which is potentially very useful and cost-effective for mobile robot applications is sonar. Since acoustic sensors are light, robust and inexpensive devices, they are widely used in applications such as navigation of autonomous vehicles through unstructured environments [4, 5], map-building [6, 7], target-tracking [8] and obstacle avoidance [9]. Although there are difficulties in the interpretation of sonar data due to multiple specular reflections, poor angular resolution of sonar and establishing correspondence between multiple echoes on different receivers [10], these difficulties can be overcome by employing accurate physical models for the reflection of sonar. The most popular sonar ranging system is based on the *time-of-flight* (TOF) measurement which is the time elapsed between the transmission of a pulse and its reception [11]. Since the amplitude of sonar signals is very prone to environmental conditions and since standard electronics for the Polaroid sensor [12] does not provide the echo amplitude directly, most sonar systems exploit only TOF information. However, for better classification of some common target primitives, multi-transducer pulse/echo systems can be employed which rely on *both* amplitude and TOF information to differentiate these targets. In earlier work by Barshan and Kuc, both TOF and amplitude of the echo signal is used to differentiate planes and corners using a statistical error model for the noisy signals [13]. Here we extend this work to develop algorithms which cover additional target types and data fusion techniques. Differential models of planes, edges, corners and cylinders have also been used by other researchers in target tracking applications [14]. In most navigation systems including sonar, two-dimensional sensing models are employed where it is assumed that targets extend into the third dimension with the same horizontal cross-section. However, it is not the case that the targets are in the same horizontal plane of the sensor in general. This problem has been handled by some researchers by providing the sensor with the flexibility to track 3-D targets which are not necessarily in the same horizontal plane as the sensor [15, 16, 17].

Sensory information from a *single* sonar has poor angular resolution and is not sufficient to differentiate some target primitives such as plane and corner [13]. Many researchers employ multiple sensor configurations in their application which can be of the same type, or of different types complementary in nature. Flynn proposed a model for combining the information from sonar and infrared sensors to build more accurate maps of the environment, in particular door edges, where sonar may fail due to its low resolution [18]. Durrant-Whyte modeled information obtained from multiple sensors to deal effectively with cooperative, competitive, and complementary interactions between various information sources [19]. A neural network model is proposed for sensory fusion based on the target localization system of the barn owl [20]. To overcome the problems that originate from the difficulties in the interpretation of sonar data for acoustic imaging, a system which combines acoustic holography with multilayer feedforward neural networks for 3-D object recognition is proposed in [21]. Neural networks are also used in sonar applications [22] and identification of ships from observed parametric radar data [23]. A comparison between neural networks and standard classifiers for radar-specific emitter identification is provided by Willson [24].

The objective in this thesis is to fuse information from uncertain environmental data acquired by multiple sonars at distinct geographical locations for strategic target recognition using two non-parametric methods. First, the ultrasonic reflection process from commonly encountered target primitives is modeled such that sonar pairs became evidential logical sensors. Logical sensors, as opposed to physical sensors that *acquire* real data, *process* real sensory data in order to generate perception units which are context-dependent interpretations of the real data [25]. By processing the real data, logical sensors classify the target primitives which are limited to plane, corner and acute corner initially. An automated perception system for mobile robots fusing uncertain sensory information must be reliable in the sense that it is predictable. Therefore quantitative approaches to uncertainty are needed. These considerations favor measure-based methods handling sensory data (both physical and logical) at different levels of granularity related to the resolution of the data as well as the time constants of the different sensors. This desire motivates our attempt to abstract the sensor integration problem in a conceptual model where uncertainty about evidence and knowledge can be measured and systematically reduced. To overcome the vulnerability of echo amplitude to noise, multiple sensors are used in the decision making, and as a first attempt, belief function approach is employed. Decisions of these sensing units are then integrated using Dempster-Shafer rule of combination. In the second approach, artificial

neural networks are employed to reduce the effect of non-parametric uncertainty in the model of ultrasonic reflection process and increase the number of target primitives which can be identified.

Chapter 2 explains the sensing configuration used in this study and introduces the target primitives. A differentiation algorithm which is employed to identify the target primitives with the measurements of a sensor pair is also provided in this chapter.

In Chapter 3, beliefs are assigned to plane, corner and acute corner based on both TOF and amplitude characteristics of the data. This section also includes the feature fusion of multiple sonars which forms the basis of logical sensing, analyzes the consensus of multiple sensors obtained by using the Dempster-Shafer rule of combination and its sensitivity to changing values of signal-to-noise ratio.

In Chapter 4, real data collected from these targets are given. Simulation results in Chapter 3 are also supported by these real data.

In Chapter 5, neural networks are reviewed briefly and the classification process with these networks are introduced. The sensitivity of neural network classifiers to changing values of signal-to-noise ratio is investigated.

In Chapter 6, discussion and concluding remarks are given, and directions for future research are motivated.

In Appendix A, geometrical models of target primitives with corresponding echo signal models are provided. Appendix B investigates different values of the noise multiplicity factor k and its choice in the classification algorithm developed in Chapter 2. The coefficients of the polynomial which is used to estimate range for an acute corner type target primitives are given in Appendix C. The amplitude characteristics of an edge with various angle values and that of a cylinder with various radius values are provided in Appendix D. In Appendix E, the nonlinear relationship between amplitude noise standard deviation and standard deviation in TOF estimates is investigated. Computer programs which are used in this study are provided in Appendix F.

Chapter 2

SONAR SENSING AND TARGET DIFFERENTIATION ALGORITHM

In this chapter, basic principles of sonar sensing are introduced. The sensing configuration and the target primitives which are used in this study are described. A differentiation algorithm is developed to identify and locate the target primitives from the measurements of a single sonar pair.

2.1 Physical Reflection Models of Sonar from Different Target Primitives

The most popular sonar ranging system is the TOF system. In this system, an echo is produced when the transmitted pulse encounters an object and a range value r is generated when the echo amplitude waveform first exceeds a preset threshold level τ (Figure 2.1):

$$r = \frac{ct_o}{2} \tag{2.1}$$

Here t_o is the TOF of the echo signal at which the echo amplitude first

exceeds the threshold level and c is the speed of sound in air.[1] Assuming Gaussian-distributed noise, τ is usually set equal to 4−5 times the value of the noise standard deviation of which can be estimated based on experimental data.

In this study, far-field model of a piston type transducer having a circular aperture is used [26]. The amplitude of the echo decreases with *inclination angle* θ, which is the deviation angle from normal incidence as illustrated in Figure 2.2. The echo amplitude falls below the threshold level when the inclination angle θ is greater than θ_o which depends on the aperture size and the resonant frequency of the transducer by:

$$\theta_o = \sin^{-1}\left(\frac{0.61c}{af_o}\right) \tag{2.2}$$

where a is the transducer aperture radius and f_o is its resonant frequency [13].

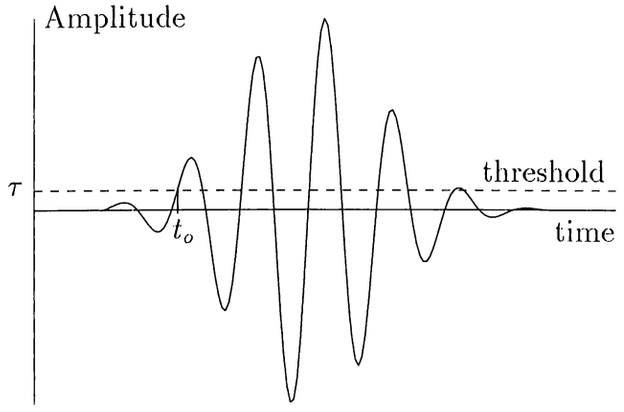


Figure 2.1: A typical echo of the ultrasound ranging system.

With a single transducer, it is not possible to estimate the azimuth of a target with better resolution than the angular resolution of sonar which is approximately $2\theta_o$. In our system, two identical acoustic transducers a and b with center-to-center separation d are employed to improve the angular resolution (Figure 2.2). Each transducer can operate both as transmitter and receiver, due to the reciprocity principle [27]. The typical shape of the *sensitivity region* of an ultrasonic transducer pair is given in Figure 2.2. The extent of this region is in general different for each target type since targets with different features exhibit different reflectivities, cross-sections and reflection characteristics.

The pressure amplitude pattern of the transducer in the far zone is well

[1] $c = 331.4\sqrt{\frac{T}{273}}$ m/s, where T is absolute temperature in Kelvin. At room temperature ($T = 293$K), $c = 343.3$ m/s.

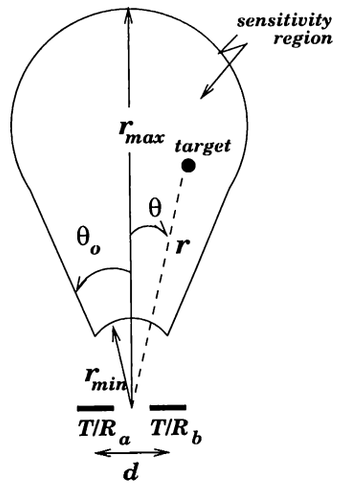


Figure 2.2: Sensitivity region of an ultrasonic transducer pair.

approximated by a Gaussian function centered around zero with standard deviation $\sigma_T = \frac{\theta_o}{2}$ [13]:

$$P(\theta) = \frac{p_o r_{min}}{r} e^{-\frac{\theta^2}{2\sigma_T^2}} \quad \text{for} \quad r \geq r_{min} \tag{2.3}$$

Moreover, for a point target the amplitude of the detected signal is the product of the beam patterns of the transmitter and receiver:

$$A(\theta_1, \theta_2) = \rho \frac{A_{max} r_{min}^2}{r_1 r_2} e^{-\frac{2(\theta_1^2+\theta_2^2)}{2\sigma_T^2}} \quad \text{for} \quad r_1, r_2 \geq r_{min} \tag{2.4}$$

where A_{max} is the maximum amplitude obtained when $\theta_1 = \theta_2 = 0^o$ and $r_1 = r_2 = r_{min}$. Here, θ_1 and θ_2 are the respective inclination angles of transmitter and receiver, and r_1 and r_2 are the distances from the target as illustrated in Figure 2.3 [13].

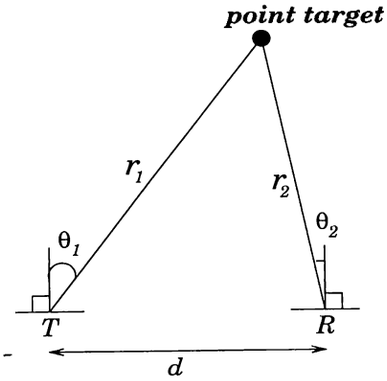


Figure 2.3: Configuration of the transducer pair with respect to a point target.

A point target in 2-D corresponds to a line target which could be a very thin cylinder or a very sharp edge target. In 3-D, it could be approximated

by a very small sphere. In this study, the target primitives employed are plane, corner, acute corner, edge and cylinder whose horizontal cross-sections are illustrated in Figure 2.4. The word *target* is used here to refer to any environment feature which is capable of being observed by a sonar sensor. These target primitives constitute the basic building blocks for most of the surfaces likely to exist in a robot's environment. Since the wavelength of sonar ($\lambda \cong 7.0$ mm at 49.4 kHz) is much larger than the typical roughness of object surfaces encountered in laboratory environments, targets in these environments reflect acoustic beams specularly like a mirror [5]. Hence, while modeling the received signals from these targets, all reflections are considered to be specular which allows transducers both transmitting and receiving to be viewed as a separate transmitter T and virtual receiver R in all cases [5].

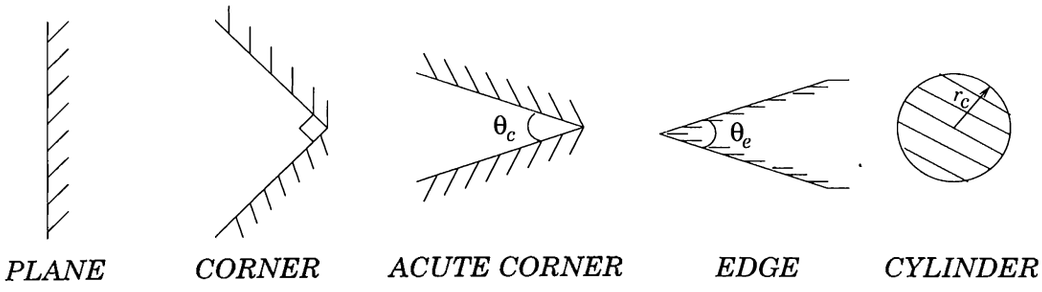


Figure 2.4: Target primitives modeled and differentiated in this study.

Detailed physical reflection models of these target primitives with corresponding echo signals are provided in Appendix A.

2.2 Target Differentiation Algorithm

In the differentiation of target primitives which are introduced in the previous section, both TOF and amplitude characteristics must be used. In Figure 2.5 and 2.6, TOF characteristics for each primitive target are given over the range of $-\theta_o < \theta < \theta_o$ where θ_o is taken as 15°. Note that the TOF characteristics of plane, corner, edge and cylinder have the same form. Initially, to differentiate an acute corner from the rest of the targets, these characteristics are used in Algorithm 1 which is introduced next. In the following, Algorithm 1 is based on TOF comparison whereas Algorithms 2 and 3 are based on amplitude comparison of the return signals.

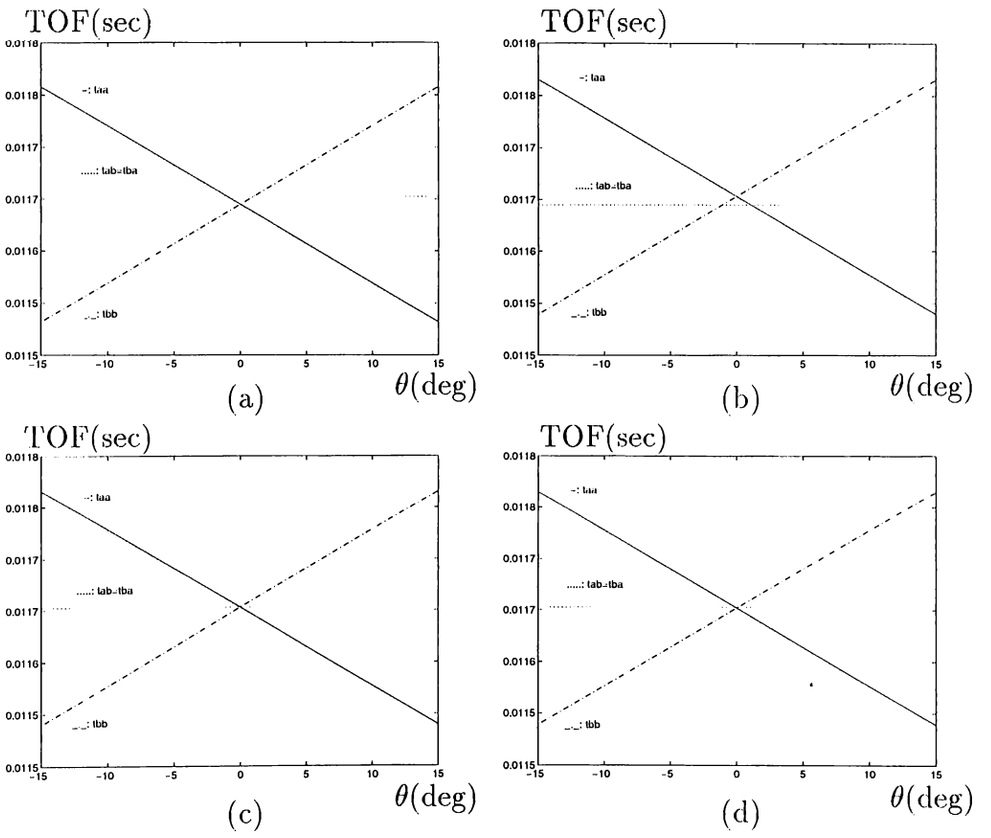


Figure 2.5: The TOF characteristics of targets when the target is at $r = 2$ m (a) plane (b) corner (c) edge with $\theta_e = 90^o$ (d) cylinder with $r_c = 20$ cm.

Algorithm 1: **Acute Corner Differentiation:**

- if $[t_{aa}(\theta) - t_{ab}(\theta)] > k_t\sigma_t$ **and** $[t_{bb}(\theta) - t_{ab}(\theta)] > k_t\sigma_t$

then *acute corner.*

- if $[t_{ab}(\theta) - t_{aa}(\theta)] > k_t\sigma_t$ **or** $[t_{ab}(\theta) - t_{bb}(\theta)] > k_t\sigma_t$

then *plane*, *corner*, *edge* or *cylinder.*

where $t_{ab}(\theta)$ denotes the TOF reading extracted at angle θ from $A_{ab}(r, \theta, d, t)$ which is the signal transmitted by a and received by b.

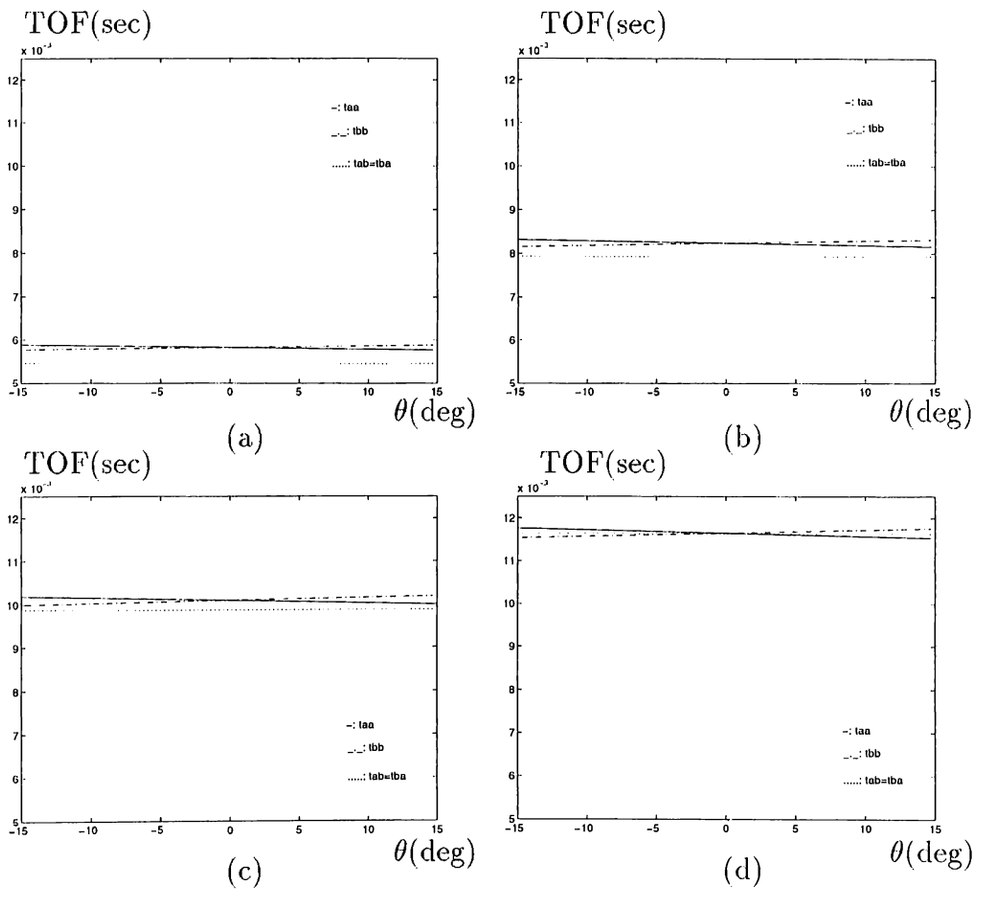


Figure 2.6: TOF characteristics of acute corner at $r = 2$ m with (a) $\theta_c = 30^o$ (b) $\theta_c = 45^o$ (c) $\theta_c = 60^o$ (d) $\theta_c = 90^o$.

In this algorithm, σ_t is the standard deviation of the TOF estimate which is in general nonlinearly related to the additive noise on the signal amplitude. This relationship is investigated in Appendix E. A multiple of σ_t, $k_t\sigma_t$, is used to improve the robustness of the differentiation algorithm to noise.

The azimuth θ and angle θ_c of an acute corner can be estimated as:

$$\theta = \sin^{-1}\left[\frac{(r_{bb}^2 - r_{aa}^2)(2r^2 + \frac{d^2}{2})}{2dr(r_{bb}^2 + r_{aa}^2)}\right] \tag{2.5}$$

$$\theta_c = \sin^{-1}\left[\sqrt{\frac{r_{aa}^2 + r_{bb}^2}{4(2r^2 + \frac{d^2}{2})}}\right] \tag{2.6}$$

where the geometry for r_{aa} and r_{bb} are given in Appendix A.

For $\theta = 0^o$:

$$r = \sqrt{\frac{d^2(r_{ab}r_{bb} + d^2 - r_{ab}^2 - r_{bb}^2)}{4(r_{ab} - r_{bb})^2 - d^2}} \tag{2.7}$$

To estimate range r for $\theta \neq 0^o$, a second-order polynomial must be solved:

$$Ax^2 + Bx + C = 0 \qquad \text{where} \qquad x = 2r^2 + \frac{d^2}{2} \tag{2.8}$$

The coefficients of this polynomial are functions of r_{aa}, r_{bb}, r_{ab}, r_{ba} and d given in Appendix C.

Note that when angle θ_c is close to 90^o, it is not possible to differentiate an acute corner by this algorithm. This algorithm can be used safely when θ_c is in the range $[0^o, 60^o]$.

In the identification of the rest of the targets, amplitude characteristics given in Figure 2.7 must be used since their TOF characteristics have the same form. Amplitude characteristics of edges with various θ_e and cylinders with various r_c are given in Appendix D. Based on the amplitude characteristics, Algorithm 2 can be used to differentiate the planar target from the rest of the target primitives.

Algorithm 2: **Plane Differentiation:**

- if $[A_{aa}(\theta) - A_{ab}(\theta)] > k_A\sigma_A$ **and** $[A_{bb}(\theta) - A_{ab}(\theta)] > k_A\sigma_A$

then *plane* with

$$r = \frac{r_a + r_b}{2} \tag{2.9}$$

$$\theta = \sin^{-1}\left(\frac{r_b - r_a}{d}\right) \tag{2.10}$$

- if $[A_{ab}(\theta) - A_{aa}(\theta)] > k_A\sigma_A$ **or** $[A_{ab}(\theta) - A_{bb}(\theta)] > k_A\sigma_A$

then *corner*, *edge* or *cylinder*.

Here $A_{ab}(\theta)$, $A_{aa}(\theta)$ and $A_{bb}(\theta)$ denote the maximum values of $A_{ab}(r, \theta, d, t)$, $A_{aa}(r, \theta, d, t)$, $A_{bb}(r, \theta, d, t)$ at angle θ, whose functional forms are provided in Appendix A. The r_a and r_b are the perpendicular distances of the respective transducers from the target whose geometry is also illustrated in Appendix A.

In this algorithm, different values for k_A have been tested and the results are presented in Appendix B. Based on these results, it is appropriate to choose $k_A = 1$.

In order to differentiate corner target primitive from edge and cylinder, amplitude measurements over the range $-\theta_o < \theta < \theta_o$ must be taken. Since the maxima of $A_{aa}(\theta)$, $A_{ab}(\theta)$, $A_{bb}(\theta)$ are the same for a right-angle corner, this type of target can be differentiated from edge and cylinder by using Algorithm 3.

Algorithm 3: **Corner Differentiation:**

• if $[\max\{A_{aa}(\theta)\} - \max\{A_{bb}(\theta)\} < \sigma_A]$ **and** $[\max\{A_{bb}(\theta)\} - \max\{A_{ab}(\theta)\} < \sigma_A]$

then *corner* with

$$r = \frac{r_a^2 + r_b^2 - \frac{d^2}{2}}{2} \tag{2.11}$$

$$\theta = \sin^{-1}\left(\frac{r_b^2 - r_a^2}{2dr}\right) \tag{2.12}$$

else *edge* or *cylinder*.

In the above algorithm, $\max\{A_{aa}(\theta)\}$ corresponds to the maximum amplitude over the range $-\theta_o < \theta < \theta_o$. With the given number of measurements, it is not possible to determine the orientation of the two planes forming the corner. Only the orientation of the line where the two planes intersect can be found with respect to the line-of-sight. To find the orientation of the planes, measurements which include reflections from the two constituent planes are necessary.

Referring to Figure 2.7, edge and cylinder targets can be distinguished over a small interval near $\theta = 0°$. At $\theta = 0°$, $A_{aa}(0) = A_{ab}(0) = A_{bb}(0)$ for an edge but this equality is not true for a cylinder. Depending on the radius of the cylinder, it may be possible to differentiate edge and cylinder with this configuration of transducers. An edge is a target with zero radius of curvature. For the cylinder, the radius of curvature has two limits of interest. As $r_c \to 0$ the characteristics of the cylinder approaches that of an edge. On the other hand as $r_c \to \infty$, the characteristics is more similar to that of a plane. By assuming the target is a cylinder first and estimating its radius of curvature [28] it may be possible to distinguish these two targets for relatively large values

of r_c. For an edge, expressions for range r and azimuth θ given in Equations 2.11 and 2.12 are the same as in the case of a corner. In the case of cylinder, in addition to range and azimuth, the radius of cylinder can also be estimated. These estimates are given by:

$$r \cong \frac{\sqrt{(r_1+r_2)^2-d^2}}{2} \tag{2.13}$$

$$\theta \cong \sin^{-1}\left[\frac{(r_a^2-r_b^2)+2r_c(r_a-r_b-d)}{2\sqrt{(r_1+r_2)^2-d^2}}\right] \tag{2.14}$$

$$r_c \cong \frac{\frac{(r_1+r_2)^2}{2}-(r_a^2+r_b^2)}{2\left(r_b+r_a-\sqrt{(r_1+r_2)^2-d^2}\right)} \tag{2.15}$$

In these equations, an approximation is used which is similar to the approximation used by Peremans *et al* in [29].

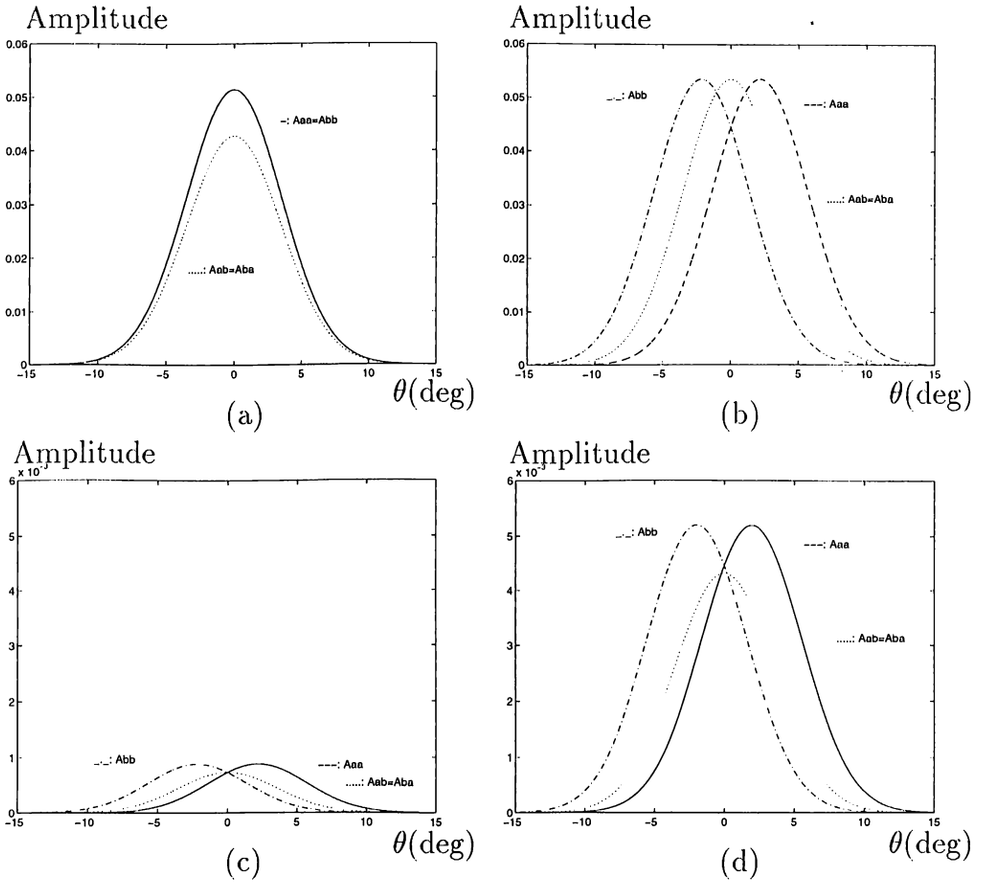


Figure 2.7: Amplitude characteristics when the targets are at $r = 2$ m(a) plane (b) corner (c) edge with $\theta_e = 90^o$ (d) cylinder with $r_c = 20$ cm.

The ratio of transducer separation to operating range (d/r) is an important parameter in the differentiation of target primitives directly affecting how well these target primitives can be differentiated by their TOF and amplitude characteristics. The further apart are the transducers, larger are the differentials in TOF and amplitude as long as the sensitivity patterns of the targets overlap. This supports improved differentiation of the targets. In the limit as $\frac{d}{r} \rightarrow 0$ which corresponds to the case that either the transducers are too close together or the target is too far, the transducers behave as a single transducer and the differential signals are not reliable. This situation is equivalent to the case of trying to differentiate the targets using a single transducer at a fixed location, which is not feasible. The effect of transducer separation d and range r on the maximum differentials is provided in Appendix A.

In this chapter, basic concepts of sonar sensing and target primitives used in this study are introduced. TOF and amplitude characteristics of each target types are given and differentiation algorithms are developed. In the next chapter, these algorithms will be used by multiple decision-making sonars to identify and locate an unknown target.

Chapter 3

LOGICAL SENSING AND FEATURE FUSION OF MULTIPLE SENSOR PAIRS

This chapter focuses on the development of a logical sensing module that produces evidential information from uncertain and partial information obtained by multiple sonars at geographically different sensing sites. The formation of such evidential information is accomplished with the theory of belief functions. As a first attempt, belief values are generated for each sensor pair and assigned to the detected features: plane and corner. These features and their evidential metric obtained from multiple sonars are then fused using the Dempster-Shafer rule of combination. Acute corner is added as an additional feature in the belief function approach and simulation results are presented.

3.1 Logical Sensing

Belief function is a mapping from a class of sets to the interval [0,1] that assigns numerical degrees of support based on evidence [30]. This is a generalization of probabilistic approaches since one is allowed to model ignorance about a given situation. Unlike probability theory, a belief function brings a metric to the intuitive idea that a portion of one's belief can be committed to a set but need not be also committed to its complement. In the target classification problem, ignorance corresponds to not having any information on the type of target that

the transducer pair is scanning.

To differentiate the target primitives, differences in the reflection characteristics of these targets are exploited and formulated in terms of degrees of belief. This logical sensor model of sonar perception is novel in the sense that it models the uncertainties associated with the target type, its range and azimuth as detected by each sensor pair. The uncertainty in the measurements of each sonar pair is represented by a belief function having target type or $feature$, and target location r and θ as focal elements with degrees of belief $b(.)$ associated with them:

$$BF = \{feature, r, \theta\,;\, b(feature), b(r), b(\theta)\} \tag{3.1}$$

3.1.1 Feature Fusion from Multiple Sonars for Plane-Corner Identification

In this study, we first focus on differentiation of two of the target types introduced, namely plane and corner. In section 3.1.3, the method will be extended to include the acute corner.

Logical sensing of target primitives is accomplished through a metric as degrees of belief assigned to plane and corner according to the amplitude and TOF characteristics of the received signals described in the previous chapter. Based on these characteristics, belief assignment to the feature focal element is made as follows

$$b(p) = I_1 \frac{[A_{aa}(\theta) - A_{ab}(\theta)][A_{bb}(\theta) - A_{ab}(\theta)]}{\max[A_{aa}(\theta) - A_{ab}(\theta)]\max[A_{bb}(\theta) - A_{ab}(\theta)]} \tag{3.2}$$

$$b(c) = \begin{cases} \frac{I_2[A_{ab}(\theta)-A_{aa}(\theta)]+I_3[A_{ab}(\theta)-A_{bb}(\theta)]}{I_2\max[A_{ab}(\theta)-A_{aa}(\theta)]+I_3\max[A_{ab}(\theta)-A_{bb}(\theta)]} & \text{if } I_2 \neq 0 \text{ or } I_3 \neq 0 \\ 0 & \text{if } I_2 = I_3 = 0 \end{cases} \tag{3.3}$$

where $b(p)$ and $b(c)$ are beliefs assigned to plane and corner features respectively and I_1, I_2 and I_3 are the indicators of the conditions given below:

$$I_1 = \begin{cases} 1 & \text{if } [A_{aa}(\theta) - A_{ab}(\theta)] > k_A\sigma_A \textbf{ and } [A_{bb}(\theta) - A_{ab}(\theta)] > k_A\sigma_A \\ 0 & \text{otherwise} \end{cases} \tag{3.4}$$

$$I_2 = \begin{cases} 1 & \text{if } [A_{ab}(\theta) - A_{aa}(\theta)] > k_A\sigma_A \\ 0 & \text{otherwise} \end{cases} \tag{3.5}$$

$$I_3 = \begin{cases} 1 & \text{if } [A_{ab}(\theta) - A_{bb}(\theta)] > k_A\sigma_A \\ 0 & \text{otherwise} \end{cases} \tag{3.6}$$

Remaining belief is assigned to an unknown target, representing ignorance, as:

$$b(u) = 1 - [b(p) + b(c)] \tag{3.7}$$

Decision of multiple sensor pairs are fused with Dempster-Shafer rule of combination [30]. Consider belief functions obtained from two sonar pairs as independent sources of evidence (Table 3.1):

$$BF_1 = \{f_i, b(f_i)\}_{i=1}^{3} = \{p, c, u; b(p), b(c), b(u)\} \tag{3.8}$$

$$BF_2 = \{g_j, b(g_j)\}_{j=1}^{3} = \{p, c, u; b(p), b(c), b(u)\} \tag{3.9}$$

Consensus is obtained as the orthogonal sum:

$$BF = BF_1 \oplus BF_2 = \{h_k, b(h_k)\}_{k=1}^{3} = \{p, c, u; b_c(p), b_c(c), b_c(u)\} \tag{3.10}$$

which is both associative and commutative. The sequential combination of multiple bodies of evidence can be obtained for n sensor pairs as:

$$BF = (((BF_1 \oplus BF_2) \oplus BF_3) \ldots \oplus BF_n) \tag{3.11}$$

Using the Dempster-Shafer rule of combination:

$$b(h_k) = \frac{\sum\sum_{h_k = f_i \cap g_j} b(f_i)b(g_j)}{1 - \sum\sum_{h_k = f_i \cap g_j = \emptyset} b(f_i)b(g_j)} \tag{3.12}$$

where $\sum\sum_{h_k = f_i \cap g_j = \emptyset} b(f_i)b(g_j)$ is a measure of conflict.

BF_1 / BF_2	plane $b_1(p)$	corner $b_1(c)$	unknown $b_1(u)$
plane $b_2(p)$	plane $b_1(p)b_2(p)$	$\emptyset$ $b_1(c)b_2(p)$	plane $b_1(u)b_2(p)$
corner $b_2(c)$	$\emptyset$ $b_1(p)b_2(c)$	corner $b_1(c)b_2(c)$	corner $b_1(u)b_2(c)$
unknown $b_2(u)$	plane $b_1(p)b_2(u)$	corner $b_1(c)b_2(u)$	unknown $b_1(u)b_2(u)$

Table 3.1: Plane/corner differentiation by Dempster-Shafer rule of combination.

Referring to Table 3.1, intersection of events plane and unknown is plane, similarly intersection of events corner and unknown is corner. The intersection of plane and corner is an empty set.

The consensus belief function representing the feature fusion is based on the features of plane, corner and acute corner. The metrics of the fusion are:

$$b(p) = \frac{b_1(p)b_2(p) + b_1(p)b_2(u) + b_1(u)b_2(p)}{1 - \text{conflict}} \tag{3.13}$$

$$b(c) = \frac{b_1(c)b_2(c) + b_1(c)b_2(u) + b_1(u)b_2(c)}{1 - \text{conflict}} \tag{3.14}$$

$$b(u) = \frac{b_1(u)b_2(u)}{1 - \text{conflict}} \tag{3.15}$$

In the above equations, the term "conflict" represents the disagreement in the consensus of two logical sensing units thus representing the degree of mismatch in the fusion of features perceived at two different sensing sites. The metric evaluating conflict is expressed as:

$$\text{conflict} = b_1(p)b_2(c) + b_1(c)b_2(p) \tag{3.16}$$

The beliefs are then rescaled after discounting this conflict and may be used in further data fusion processes.

3.1.2 Fusion of Range and Azimuth Estimates

Assignment of belief to range and angle measurements is based on the simple observation that the closer the target is to the face of the transducer, the more accurate is the range reading, and the closer the target is to the line-of-sight of the transducer, the more accurate is the angle estimate. This is due to the physical properties of sonar: signal amplitude decreases with r and with $|\theta|$. At large ranges and larger angular deviations, signal-to-noise ratio is smaller. Most accurate measurements are obtained along the line-of-sight ($\theta = 0^o$) and at nearby ranges to the sensor pair. Therefore, belief assignments to range and azimuth estimates derived from the TOF measurements are made as follows [31]:

$$b(r) = \frac{r_{max} - r}{r_{max} - r_{min}} \tag{3.17}$$

$$b(\theta) = \frac{\theta_o - |\theta|}{\theta_o} \tag{3.18}$$

Note that, belief of r takes its maximum value of one when $r = r_{min}$ and its minimum value of zero when $r = r_{max}$. Similarly, belief of θ is one when $\theta = 0^o$ and zero when $\theta = \pm\theta_o$.

Since each sensor pair takes measurements in own its sensor-centric coordinate frame, the beliefs of range and azimuth information need to be first

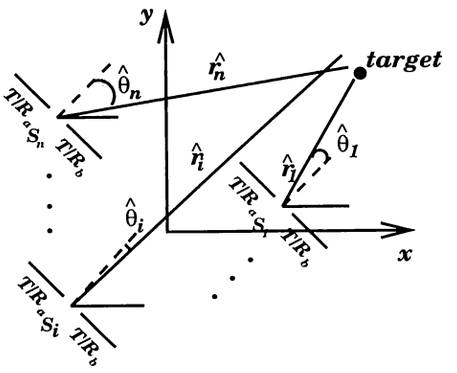


Figure 3.1: Common coordinate system for n pairs of sonar sensors.

projected onto a common coordinate system where they can be integrated. This is represented in Figure 3.1 where erroneous estimates are assumed for r and θ. Then the metric of the fusion process is computed based on these projected values. Due to the noise on the system, estimated range and azimuth values are different from the true values. Suppose n transducer pairs are employed and each pair estimates the range and azimuth of the target as:

$$(\hat{r_i}, \hat{\theta_i}) \quad i = 1, 2, \ldots, n \tag{3.19}$$

Correct position of the target is denoted by

$$(r_i, \theta_i) \quad i = 1, 2, \ldots, n \tag{3.20}$$

in each sensor's own coordinate frame while the target is within its sensitivity region.

The projected range and azimuth are represented in Figure 3.2 as:

$$(r_i^{'}, \theta_i^{'}) \quad i = 1, 2, \ldots, n \tag{3.21}$$

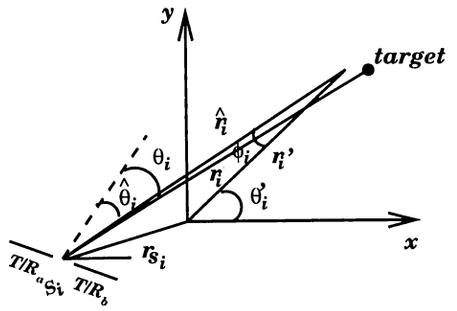


Figure 3.2: Projected range and azimuth for transducer pair i.

Although typically logarithmic relationships are used to relate uncertainty and belief [32], here we choose to use a simpler linear relationship

$$\frac{\sigma_p}{p} = \frac{1 - b(p)}{b(p)}. \tag{3.22}$$

where p corresponds to either the range or azimuth of the target.

Since the range and azimuth estimates are transformed onto a common coordinate frame, uncertainties in the estimated range and azimuth must be represented as uncertainties in the transformed range and azimuth with the transformation below:

$$\sigma_{r_i'} = \sigma_{\hat{r_i}} |\cos\phi_i| + \hat{r_i}\sigma_{\hat{\theta_i}} |\sin\phi_i| \quad (3.23)$$

$$r_i'\sigma_{\theta_i'} = \hat{r_i}\sigma_{\hat{\theta_i}} |\cos\phi_i| + \sigma_{\hat{r_i}} |\sin\phi_i| \quad (3.24)$$

where $\sigma_{\hat{r_i}}$ and $\sigma_{\hat{\theta_i}}$ represent uncertainties in the range and azimuth measurements respectively and ϕ_i is the angle between $\hat{r_i}$ and r_i'. Since the position of the i'th transducer pair, r_{s_i}, is known, ϕ_i can be found from the geometry by using the cosine theorem:

$$\phi_i = \cos^{-1}\left(\frac{\hat{r_i}^2 + r_i'^2 - r_{s_i}^2}{2\hat{r_i}r_i'}\right) \quad (3.25)$$

where r_{s_i} is the distance of i'th sensor pair from the origin. After projecting the range and azimuth estimates onto a common coordinate system, they are fused into a single range and a single azimuth estimate as follows:

$$r_f = \frac{\sum_{i=1}^{n} r_i' b(r_i')}{\sum_{i=1}^{n} b(r_i')} \quad (3.26)$$

$$\theta_f = \frac{\sum_{i=1}^{n} \theta_i' b(\theta_i')}{\sum_{i=1}^{n} b(\theta_i')} \quad (3.27)$$

where the new belief value in the common coordinate system can be found by solving Equation 3.22 for $b(p)$.

Beliefs to these combined range and azimuth estimates are assigned by using Equations 3.17 and 3.18. When the system is noiseless and the location of the target is (r, θ) in the common coordinate system, all estimated range and azimuth values are equal to their true values:

$$\hat{r_i} = r_i \quad (3.28)$$

$$\hat{\theta_i} = \theta_i \quad (3.29)$$

Then the projected and fused range and azimuth estimates are equal:

$$r_i' = r_f = r \quad (3.30)$$

$$\theta_i' = \theta_f = \theta \quad i = 1, 2, \ldots, n \quad (3.31)$$

For the planar target case, since each sensor pair detects the plane at a different position, fusion of range and azimuth estimates must be modified

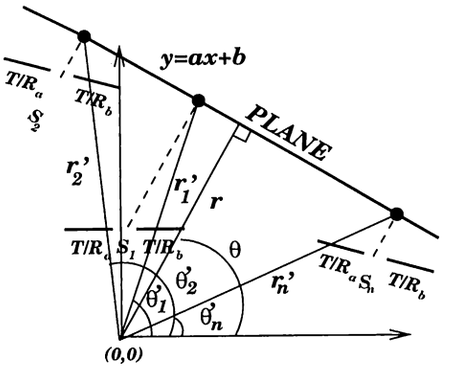


Figure 3.3: Position of a plane with respect to each sensor pair.

(Figure 3.3). For this case, a line which represents the plane in 2-D can be estimated using the estimated positions of the plane by all sensor pairs in the common coordinate frame. Then the perpendicular distance between this line and the origin and the orientation of this line with respect to the origin must be found which are the fused range and azimuth of this plane.

In 2-D, a planar target can be represented as a line with the equation:

$$y = ax + b \tag{3.32}$$

Since we have range and azimuth measurements from n sensors, we seek a weighted least squares solution for a and b where the weights and uncertainty are inversely related. The weighted least squares solution can be found by solving the following equation:

$$\min_{a,b} \sum_{i=1}^{n} w_i [y_i - (ax_i + b)]^2 \tag{3.33}$$

Here

$$x_i = r_i' \cos\theta_i' \tag{3.34}$$

$$y_i = r_i' \sin\theta_i' \quad i = 1, ..., n \tag{3.35}$$

and the weights which minimize the mean square error can be found as [13]:

$$w_i = \frac{1}{\sigma_{x_i}^2 + \sigma_{y_i}^2} \tag{3.36}$$

where σ_{x_i} and σ_{y_i} are found by transforming the uncertainties in r_i' and θ_i' as:

$$\sigma_{x_i} = \sigma_{r_i'} |\cos\theta_i'| + r_i' \sigma_{\theta_i'} |\sin\theta_i'| \tag{3.37}$$

$$\sigma_{y_i} = \sigma_{r_i'} |\sin\theta_i'| + r_i' \sigma_{\theta_i'} |\cos\theta_i'| \tag{3.38}$$

Note that here there is no need to normalize the sum of the weights to one. Then, the weighted least squares solution of this problem is:

$$a_{\text{WLS}} = \frac{\left(\sum_{i=1}^{n} w_i\right)\left(\sum_{j=1}^{n} w_j x_j y_j\right) - \left(\sum_{i=1}^{n} w_i x_i\right)\left(\sum_{j=1}^{n} w_j y_j\right)}{\left(\sum_{i=1}^{n} w_i\right)\left(\sum_{j=1}^{n} w_j x_j^2\right) - \left(\sum_{i=1}^{n} w_i x_i\right)^2} \tag{3.39}$$

$$b_{\text{WLS}} = \frac{\sum_{i=1}^{n} w_i y_i - a_{\text{WLS}} \sum_{i=1}^{n} w_i x_i}{\sum_{i=1}^{n} w_i} \tag{3.40}$$

and the fused range and azimuth estimates are:

$$r_f = r = \frac{b_{\text{WLS}}}{\sqrt{a_{\text{WLS}}^2 + 1}} \tag{3.41}$$

$$\theta_f = \theta = \tan^{-1}\left(-\frac{1}{a_{\text{WLS}}}\right) \tag{3.42}$$

3.1.3 Inclusion of Acute Corner Target Type in the Classification Process

When an acute corner target model is included in the target classification process using the belief function approach introduced above, belief to an acute corner can be assigned as:

$$b(ac) = I_4 \frac{[t_{aa}(\theta) - t_{ab}(\theta)][t_{bb}(\theta) - t_{ab}(\theta)]}{\max\{[t_{aa}(\theta) - t_{ab}(\theta)][t_{bb}(\theta) - t_{ab}(\theta)]\}} \tag{3.43}$$

where I_4 is an indicator of the condition given below:

$$I_4 = \begin{cases} 1 & \text{if } [t_{aa}(\theta) - t_{ab}(\theta)] > \sigma_t \quad \textbf{and} \quad [t_{bb}(\theta) - t_{ab}(\theta)] > \sigma_t \\ 0 & \text{otherwise} \end{cases} \tag{3.44}$$

Due to the inclusion of an acute corner, the belief assignments to plane and corner in Equations 3.2 and 3.3 must be modified by multiplying these equations by $(1 - I_4)$ such that:

$$b(p) = (1 - I_4)\, I_1 \frac{[A_{aa}(\theta) - A_{ab}(\theta)][A_{bb}(\theta) - A_{ab}(\theta)]}{\max[A_{aa}(\theta) - A_{ab}(\theta)] \max[A_{bb}(\theta) - A_{ab}(\theta)]} \tag{3.45}$$

$$b(c) = \begin{cases} (1 - I_4)\, \frac{I_2[A_{ab}(\theta) - A_{aa}(\theta)] + I_3[A_{ab}(\theta) - A_{bb}(\theta)]}{I_2 \max[A_{ab}(\theta) - A_{aa}(\theta)] + I_3 \max[A_{ab}(\theta) - A_{bb}(\theta)]} & \text{if } I_2 \neq 0 \text{ or } I_3 \neq 0 \\ 0 & \text{if } I_2 = I_3 = 0 \end{cases} \tag{3.46}$$

With this modification, when a sensor pair decides that the target is an acute corner with belief $b(ac)$, then this sensor pair automatically assigns zero belief to a plane or a corner. Again, remaining belief is assigned to an unknown target, representing ignorance as:

$$b(u) = 1 - [b(p) + b(c) + b(ac)] \tag{3.47}$$

Inclusion of other target types such as acute corner to the rule of combination with two sensor pairs is a simple extension of the combination rule in Table 3.1:

BF_1 / BF_2	plane $b_1(p)$	corner $b_1(c)$	acute corner $b_1(ac)$	ignorance $b_1(u)$
plane $b_2(p)$	plane $b_1(p)b_2(p)$	$\emptyset$ $b_1(c)b_2(p)$	$\emptyset$ $b_1(ac)b_2(p)$	plane $b_1(u)b_2(p)$
corner $b_2(c)$	$\emptyset$ $b_1(p)b_2(c)$	corner $b_1(c)b_2(c)$	$\emptyset$ $b_1(ac)b_2(c)$	corner $b_1(u)b_2(c)$
acute corner $b_2(ac)$	$\emptyset$ $b_1(p)b_2(ac)$	$\emptyset$ $b_1(c)b_2(ac)$	acute corner $b_1(ac)b_2(ac)$	acute corner $b_1(u)b_2(ac)$
ignorance $b_2(u)$	plane $b_1(p)b_2(u)$	corner $b_1(c)b_2(u)$	acute corner $b_1(ac)b_2(u)$	ignorance $b_1(u)b_2(u)$

Table 3.2: Target differentiation by Dempster-Shafer rule of combination.

In this case, the consensus belief function representing the feature fusion is:

$$b(p) = \frac{b_1(p)b_2(p) + b_1(p)b_2(u) + b_1(u)b_2(p)}{1 - \text{conflict}} \tag{3.48}$$

$$b(c) = \frac{b_1(c)b_2(c) + b_1(c)b_2(u) + b_1(u)b_2(c)}{1 - \text{conflict}} \tag{3.49}$$

$$b(ac) = \frac{b_1(ac)b_2(ac) + b_1(ac)b_2(u) + b_1(u)b_2(ac)}{1 - \text{conflict}} \tag{3.50}$$

$$b(u) = \frac{b_1(u)b_2(u)}{1 - \text{conflict}} \tag{3.51}$$

where "conflict" represents the disagreement between the decisions of the two sensor pairs:

$$\begin{aligned}\text{conflict}=& b_1(p)b_2(c) + b_1(c)b_2(p) + b_1(c)b_2(ac) \\ &+ b_1(ac)b_2(c) + b_1(ac)b_2(p) + b_1(p)b_2(ac)\end{aligned} \tag{3.52}$$

The projection of range and azimuth estimates to a common coordinate system and their combination are similar to the plane/corner case. The total belief assignment to the combined range and azimuth estimates is also the same as in the previous case.

3.2 Simulation Results

Simulations are performed with the help of C++ programming language on a SUN SPARC STATION 2 and related programs (Programs 6–8) are given in Appendix F.

3.2.1 Feature Fusion for Plane-Corner Identification

In the simulated system, it is assumed that a decision-making unit consisting of a pair of logical sensors are available with separation $d = 15.0$ cm, mounted on a stepper motor with stepping angle 0.9^o. Signals are simulated according to the models presented in Appendix A. Zero-mean additive Gaussian noise of standard deviation σ_A is added to the echo amplitude. At each step of the motor, a pulse is transmitted, four TOF and four amplitude measurements are collected. This transducer pair scans the mobile robot laboratory area which is a 5m×5m room for $-180^o \leq \phi \leq 180^o$ in order to detect corners and planar walls.

The results of belief assignment for a single transducer pair located at the center of the room are given in Figure 3.4. In this figure, $b(p)$ clearly shows that "plane" feature is recognized with high beliefs at right angles at 0^o, around -90^o, $+90^o$, $\pm 180^o$ and with highest beliefs in range since these features lie at closest proximity to the sonar. The belief $b(c)$ shows that the four corners of the room are identified with highest belief values around $\pm 45^o$ and $\pm 135^o$. The belief chop in the middle of each corner belief curve reflects a pin-type rise in uncertainty at these locations. This is due to the amplitude characteristics of the corner. At $+\varepsilon$, $-\varepsilon$ degrees to the left or to the right of this line, higher beliefs are generated in the recognition of a corner. In the angular interval between the identification of plane and that of corner in $b(u)$, there exists a region of high uncertainty that corresponds to no data acquisition. This region is approximately 22^o in which all the transmitted waveforms bounce off the room boundaries and no return signal is available, thus $b(r) = b(\theta) = 0$.

Simulations we performed further evolved in working with three transducer pairs located at different positions in the 5m×5m square room and their decisions are combined so as to generate a feature fusion by employing the Dempster-Shafer rule of combination. The locations of these transducer pairs are $(-0.5, 1)$, $(-2, 2)$ and $(2, 2)$ in meters, where the origin is taken as the center of the room. All transducer pairs are assumed to rotate on stepper

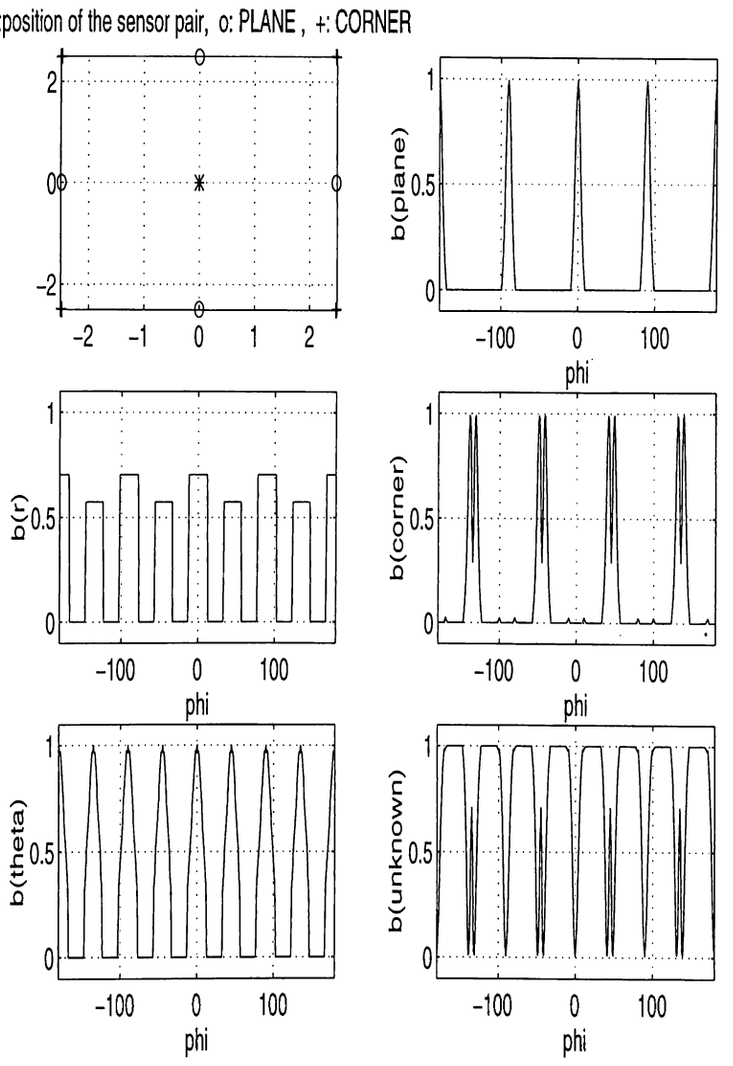


Figure 3.4: Belief assignment for a single transducer pair.

motors with stepping angle 0.9^o. These transducer pairs scan the room for $-180^o \leq \phi \leq 180^o$. At each step, transducer pairs collect data from the target at the same step angle ϕ, and the decisions of all pairs at this angle are fused. In order to calculate probabilities of correct classification, misclassification and lack of target identification, data is collected for $-180^o \leq \phi \leq 180^o$ three times which corresponds to about 1200 decisions. The classification results for a single transducer pair located at $(-0.5, 1)$ and the data fusion using three transducer pairs are given in Figures 3.5 and 3.6 respectively. In both figures, the left-hand side gives the configuration of the sensor pair(s) within the room. On the right-hand side, the probability curves of correct classification, misclassification and unknown target are plotted as functions of σ_A. The curve representing the probability of correct classification derived from the consensus of the three sonars illustrates how fusion provides an increase in evidential

support that raises the probability of correct classification when compared to that of a single transducer pair such as in Figure 3.5.

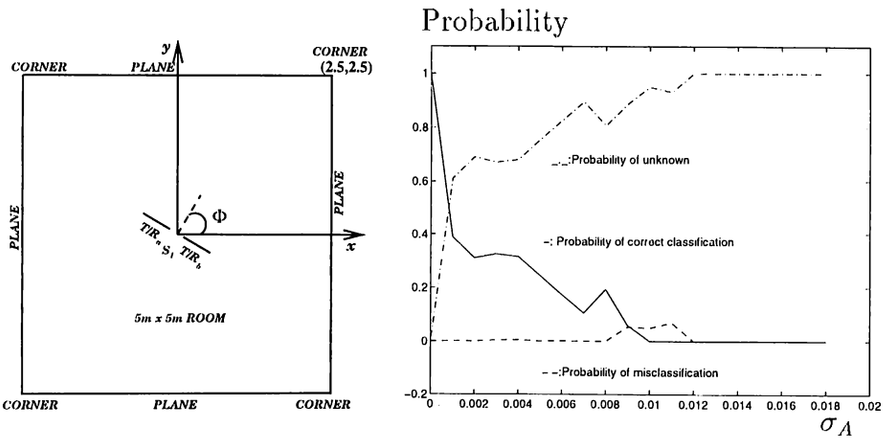


Figure 3.5: Classification with a single transducer pair.

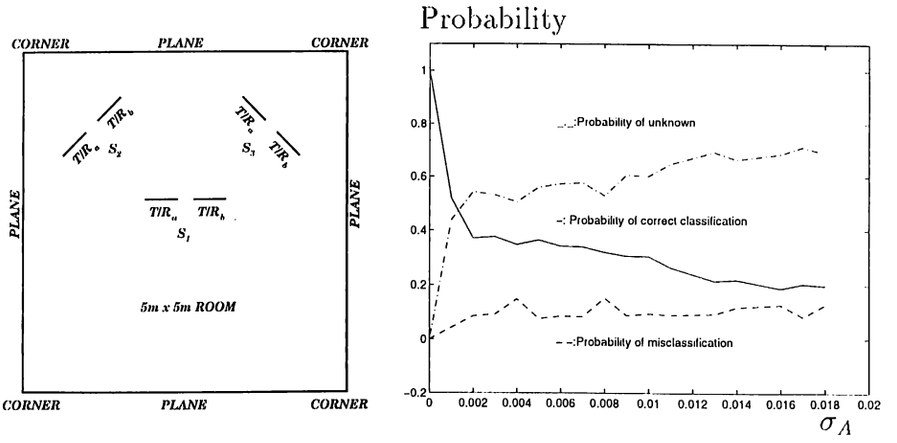


Figure 3.6: Classification and data fusion with three transducer pairs.

When the maximum signal amplitude is 0.06, amplitude noise standard deviation of 0.006 or larger corresponds to 75% of the maximum signal amplitude differences. For $\sigma_A \geq 0.006$, differential signal levels are comparable to the noise level and it becomes impossible to detect these differences. In Figure 3.5, the probability of misclassification of one pair is almost zero for all standard deviation values of the noise due to the inclusion of σ_A in the classification algorithms. In the case of replacing σ_A with zero, the probability of misclassification increases for all noise standard deviation values used in this study. In this case, the performance of the classification is comparable to the performance of a randomized decision rule [33] where 50% of the time the target is randomly guessed to be a plane, 50% of the time a corner by

completely ignoring the data. The σ_A term is included in the differentiation algorithm to provide robustness against noise on the signals as explained in detail in Appendix B. Probabilities in this case can be viewed in Figure 3.7.

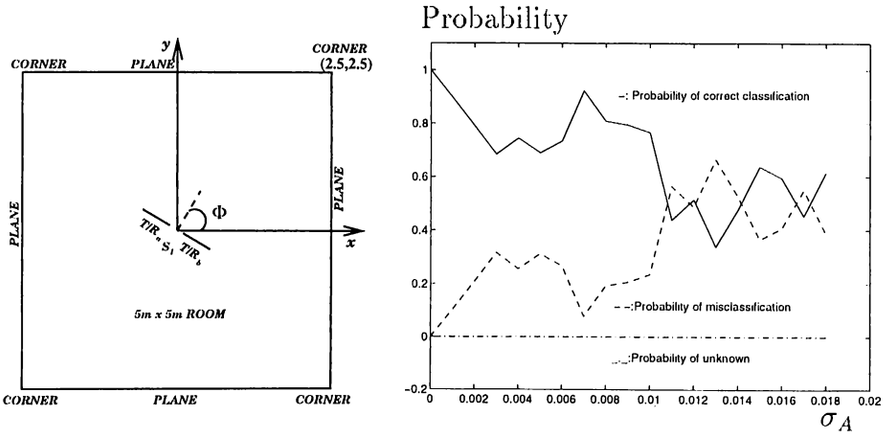


Figure 3.7: Classification with a single transducer pair without the σ_A term in the classification algorithm.

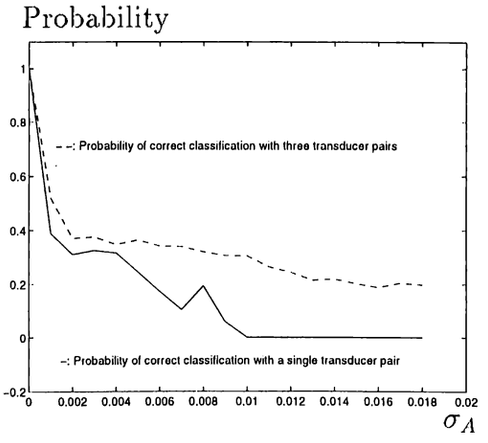


Figure 3.8: Improvement in the probability of correct classification caused by data fusion with three transducer pairs.

Although the performance for small ($\sigma_A < 0.002$) noise levels are comparable, for σ_A beyond 0.002, improvement in correct classification of the target is 20%. Of course, this is at increased cost of time to collect the data and do the computations necessary to fuse the data from three pairs of sensors. Dempster-Shafer rule of combination is computationally complex which requires approximately twice the computational effort of Bayesian inference for two or three sensors in a non-parallel implementation [34]. The improvement in the probability of correct classification with data fusion can be seen in Figure 3.8.

3.2.2 Simulation Results of Fusion of Range and Azimuth Estimates

Fusion of range and azimuth estimates is done in the same simulated room with the same three sensor pairs which are placed at the same position as in the previous section. Range and azimuth estimates are fused at $\phi = 90^o$ at which target is a plane and $\phi = 135^o$ at which target is a corner.

Simulation results of fusion of range and azimuth estimates at $\phi = 90^o$ and $\phi = 135^o$ are given in Figures 3.9–3.10 respectively. In Figure 3.9, sensor pairs which are located at (−2.0,2.0) and (2.0,2.0) respond in a similar way to the change in the amplitude noise. Their echo amplitudes from the plane are the same at $\phi = 90^o$ since they are at the same distance from this plane. However, sensor pair which is located at (−0.5,1.0) can recognize this plane up to a small noise standard deviation value of 0.002 since it is the furthest pair to this plane. After this value of amplitude noise standard deviation, echo amplitudes are comparable to noise and it becomes impossible to detect these signals. As a result, the recognition of the plane by this sensor pair is random for larger values of noise standard deviation. The maximum range and azimuth errors are about 1.2 cm and 0.2^o respectively for the two closest sensor pairs. The furthest pair does not detect the plane at some noise standard deviation values. An error minimization is obtained by the fusion of range and azimuth estimates of these sensor pairs. Therefore, the maximum errors in fused range and azimuth estimates are only 1.7 cm and 0.004^o respectively.

In Figure 3.10, sensor pair which is located at (2.0,2.0) does not have any contribution to the fused range and azimuth estimates since it does not detect the corner at $\phi = 135^o$. Moreover, the echo amplitude which is reflected from the plane is not received by this sensor pair since the inclination angle between line-of-sight of this sensor pair and the perpendicular distance to the plane is 45^o which is greater than θ_o. The sensor pair which is located at (−0.5,1.0) has a contribution to the fused range and azimuth estimates when the system is noiseless. Since the distance between this sensor pair and corner is 3.2 m at which echo amplitude which is reflected from this corner is comparable to the amplitude noise standard deviation values used in this study, it is impossible to detect difference signals which are used in the differentiation algorithm. Due to this fact, this sensor pair cannot make any decision about the target at these amplitude noise standard deviation values. Only contribution to the fusion process comes from the sensor pair which is located at (−2.0,2.0) when there exists noise in the system. This sensor pair is also the closest one to this corner.

The maximum errors in the fused range and azimuth estimates are 1.6 cm and 0.7^o respectively.

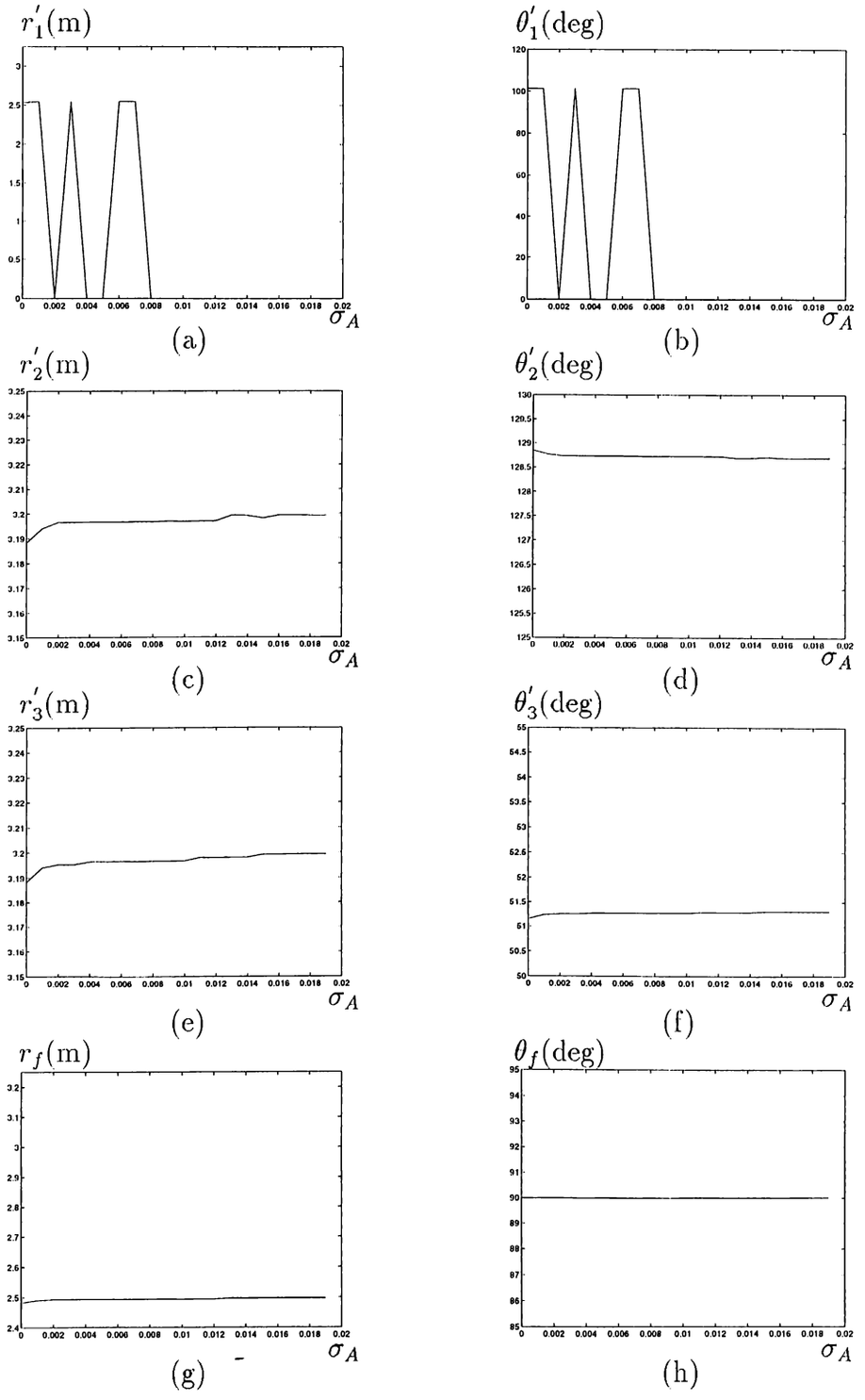


Figure 3.9: Projected range and azimuth estimates at $\phi = 90^o$ for the sensor pair which is located at (a),(b) (−0.5.1.0) (c),(d) (−2.0,2.0) (e),(f) (2.0,2.0) (g),(h) Fused range and azimuth estimates.

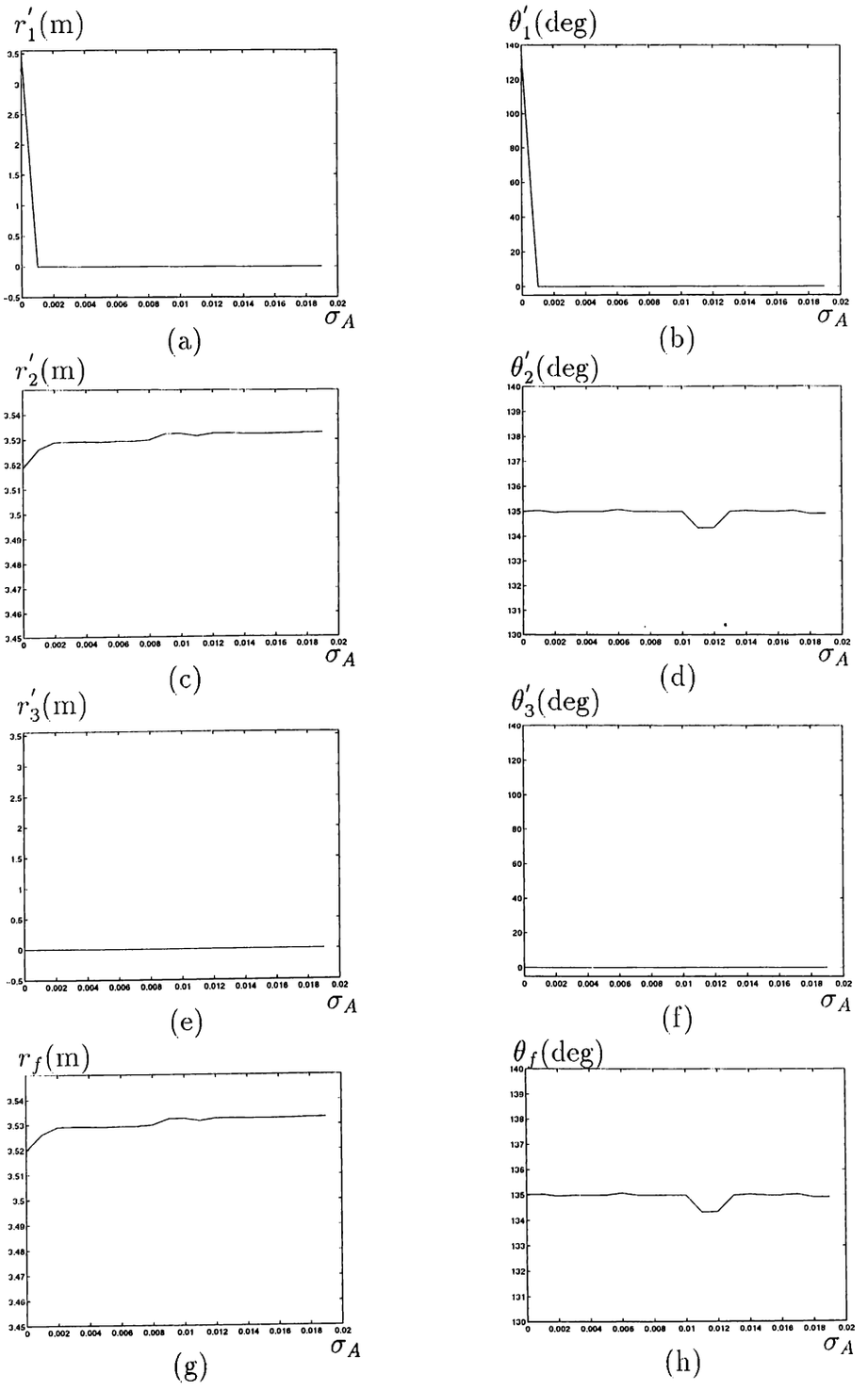


Figure 3.10: Projected range and azimuth estimates at $\phi = 135^o$ for the sensor pair which is located at (a),(b) (−0.5.1.0) (c),(d) (−2.0,2.0) (e),(f) (2.0,2.0) (g),(h) Fused range and azimuth estimates.

3.2.3 Simulation Results with Acute Corner Target Model

In the acute corner simulations, same sensing configuration is used with exactly the same parameters as in the previous section. An acute corner with wedge angle θ_c is placed in front of the sensor pair at $r = 2$ m as shown in Figure 3.11. Each time a pulse is transmitted, four TOF and four amplitude measurements are collected. The stepper motor is rotated and the target is scanned for a range of θ from -15^o to 15^o. While obtaining classification results for each case, the transducer pair scans the target from $\theta = -15^o$ to $\theta = 15^o$, 35 times. As a result, the transducer pair makes about 1200 decisions for each pair of σ_t and σ_A values.

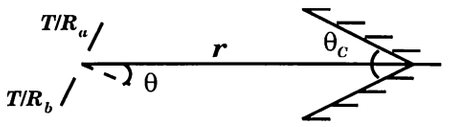


Figure 3.11: Position of the transducer pair and acute corner.

The results of belief assignments by this transducer pair for three different values of θ_c are given in Figures 3.12–3.14. Referring to these figures, maximum belief value of being an acute corner is one which is obtained at $\theta = 0^o$ for all θ_c values when the system is noiseless. Moreover, the belief of being a plane or a corner is zero for all θ, θ_c and σ_A values in these figures. The values of σ_A used in this analysis are 0.002 and 0.003. Since these beliefs are affected by σ_t and 0.68 μsec is the minimum value of σ_t obtained at $\sigma_A = 0.002$, 7.61 μsec is its maximum value when $\sigma_A = 0.003$. These values are obtained from the simulations performed to investigate the relationship between σ_t and σ_A which are given in Appendix E. The relationship between σ_t and σ_A for a threshold level $\tau = 5\sigma_A$ which is used in our all simulation studies is presented in Table 3.3. Although the decrease in the belief of being an acute corner with increasing $|\theta|$ is sharper for larger θ_c, the belief of being an acute corner is greater than the belief of being an unknown target for all θ and σ_A values taking values between 0.8 and 1 for $\theta_c = 30^o$, 0.7 and 1 for $\theta_c = 45^o$ and 0.6 and 1 for $\theta_c = 60^o$ even if $\sigma_A = 0.003$ at which σ_t takes its maximum value in the interval $\sigma_A \in [0, 0.008]$. The limit of this interval $\sigma_A = 0.008$ corresponds to the maximum difference in the echo amplitudes.

σ_A	$\sigma_t(\mu\text{sec})$
0.0	0.0
0.001	1.30
0.002	0.68
0.003	7.61
0.004	6.46
0.005	3.27
0.006	2.09
0.007	2.04
0.008	5.43

Table 3.3: Relationship between σ_A and σ_t.

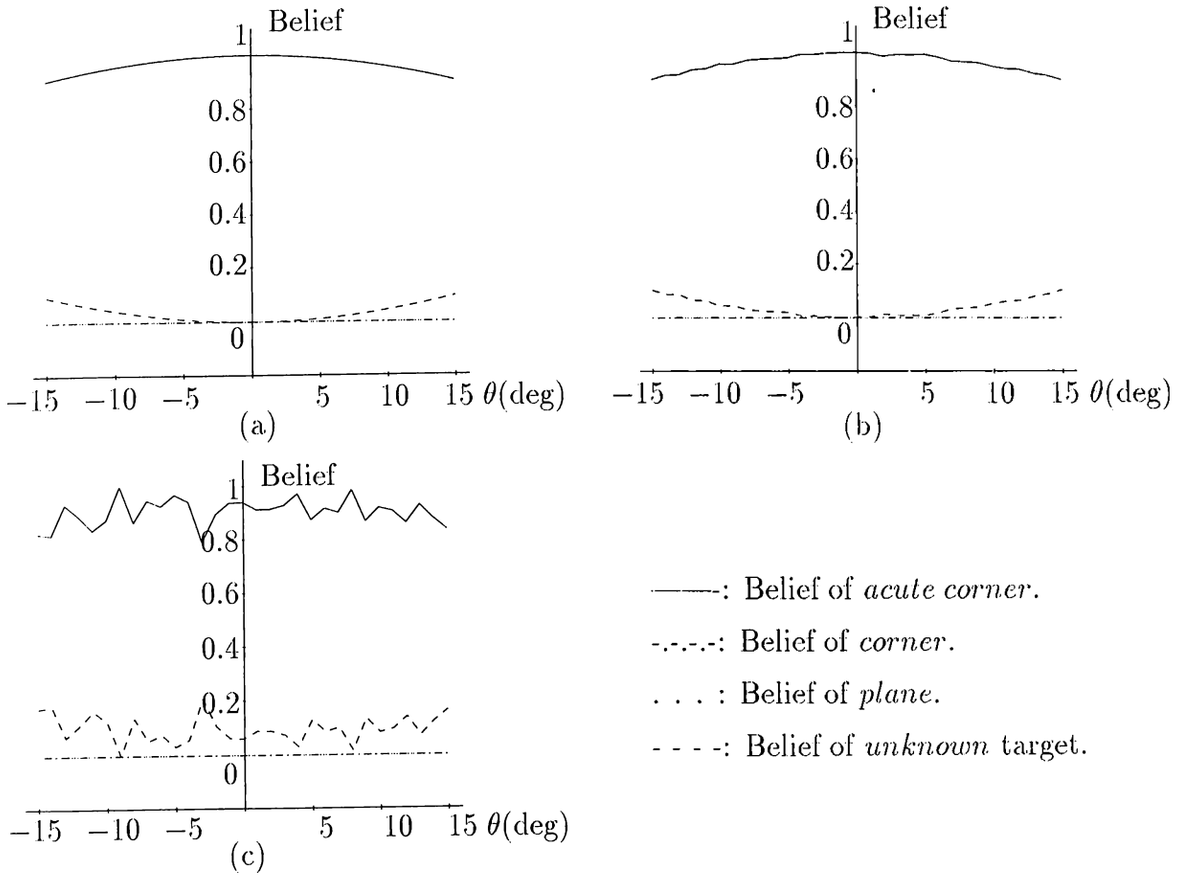


Figure 3.12: Assignment of beliefs by a transducer pair when an acute corner of $\theta_c = 30^o$ at $r = 2$ m is scanned for noise standard deviation values (a) $\sigma_A = 0.0$ (b) $\sigma_A = 0.002$ (c) $\sigma_A = 0.003$.

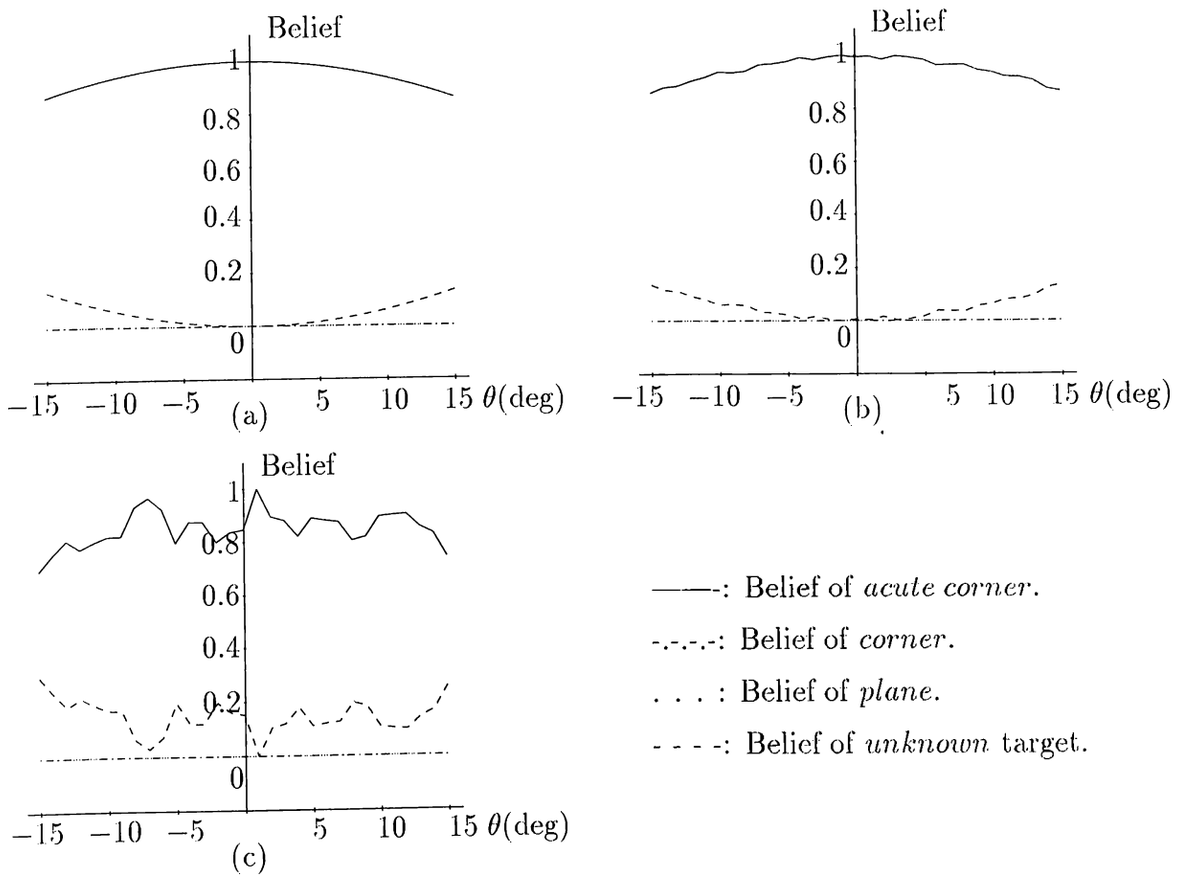


Figure 3.13: Assignment of beliefs by a transducer pair when an acute corner at $r = 2$ m with $\theta_c = 45^o$ is scanned for noise standard deviation values (a) $\sigma_A = 0.0$ (b) $\sigma_A = 0.002$ (c) $\sigma_A = 0.003$.

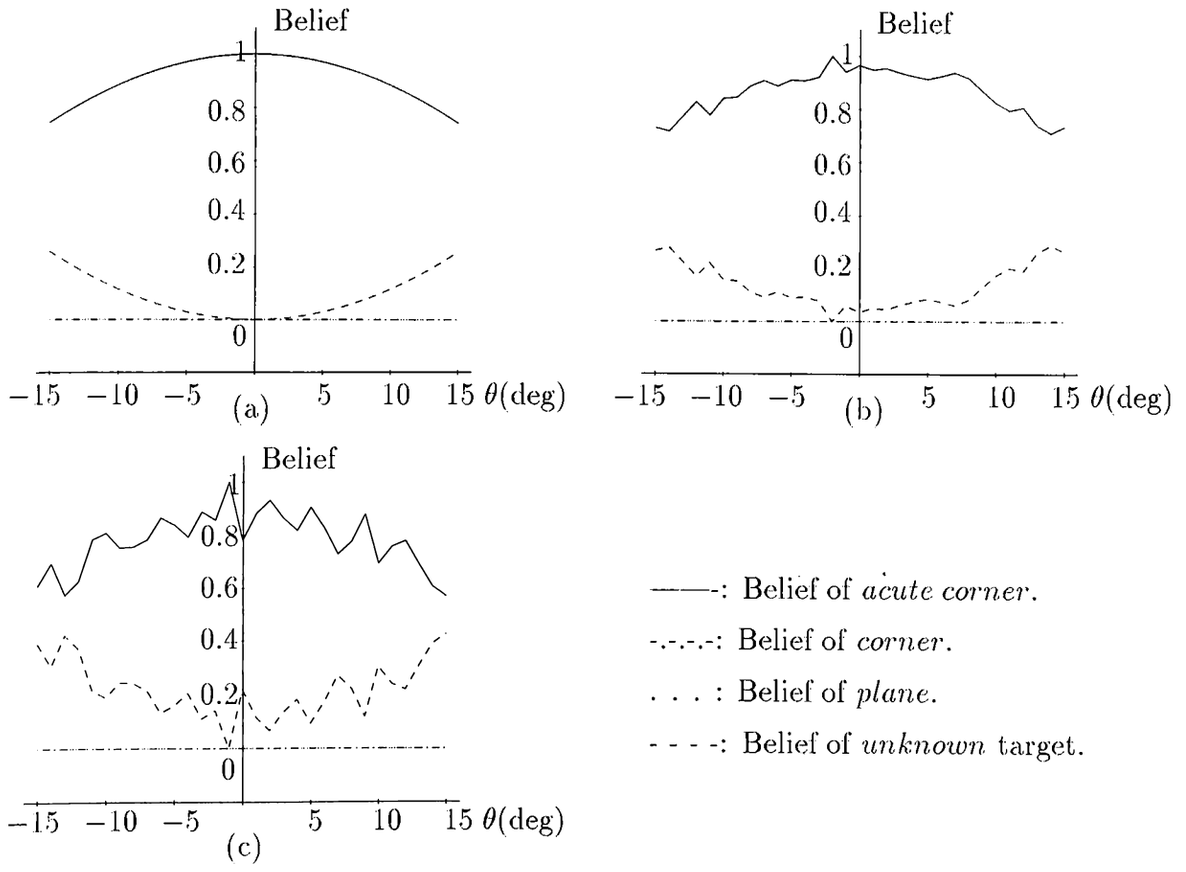
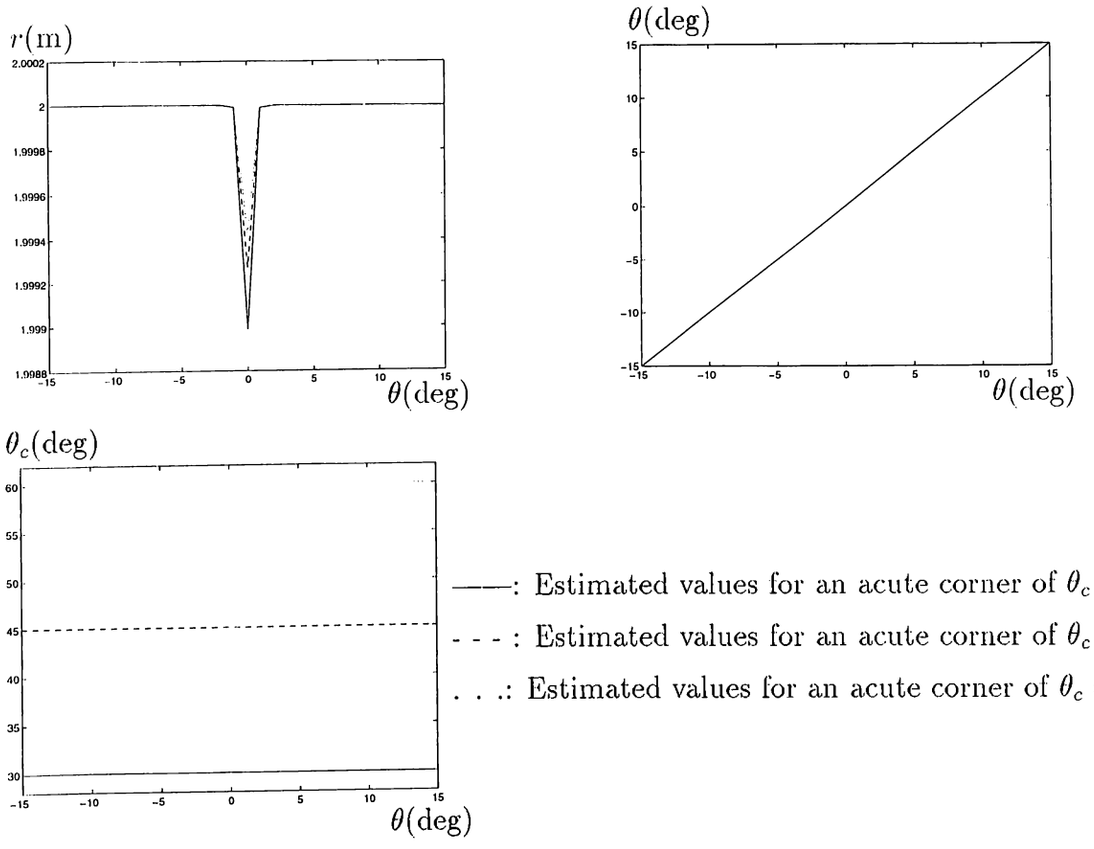


Figure 3.14: Assignment of beliefs by a transducer pair when an acute corner at $r = 2$ m with $\theta_c = 60^o$ is scanned for noise standard deviation values (a) $\sigma_A = 0.0$ (b) $\sigma_A = 0.002$ (c) $\sigma_A = 0.003$.

The range, azimuth and the angle of acute corner θ_c are estimated for these three acute corners and σ_A values which are given in Figures 3.15–3.17. In these estimations Equations 2.5–2.7 in Section 2.2 are used. Referring to Figure 3.15, maximum errors in range at $\theta = 0^o$ are 1 mm for $\theta_c = 30^o$, 0.8 mm for $\theta_c = 45^o$ and 0.6 mm for $\theta_c = 60^o$. The dip at $\theta = 0^o$ is due to the approximation done in estimation of r at $\theta = 0^o$ and errors in azimuth and θ_c are zero for all acute corners. For the $\sigma_A = 0.002$, the maximum error in range is 5.7 cm which is obtained for the acute corner of $\theta_c = 30^o$, and maximum error in azimuth is 0.8^o and θ_c is 1.7^o for the acute corner of $\theta_c = 60^o$ in Figure 3.16. For the $\sigma_A = 0.003$, the maximum error in range is 34 cm for the acute corner of $\theta_c = 30^o$, in azimuth is 4.6^o, and θ_c is 9.9^o which are obtained for the acute corner of $\theta_c = 60^o$ in Figure 3.17.

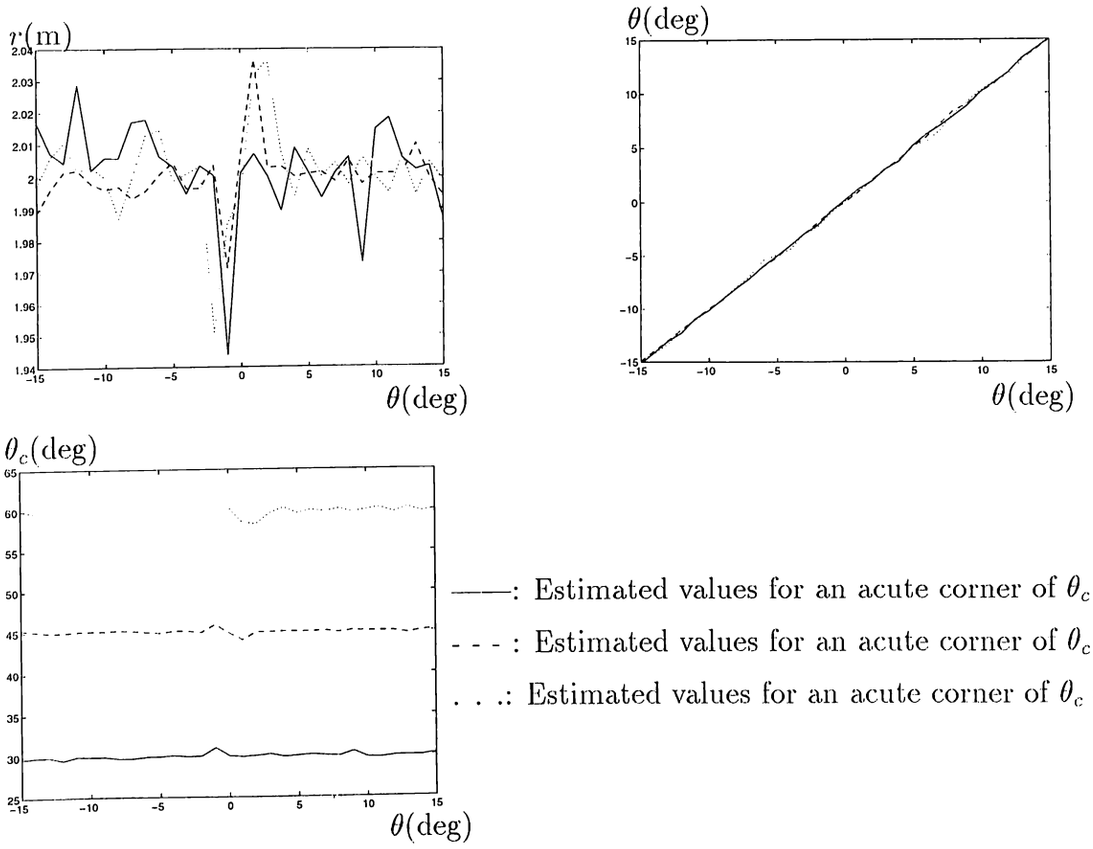


——: Estimated values for an acute corner of $\theta_c = 30^o$.

- - - : Estimated values for an acute corner of $\theta_c = 45^o$.

. . .: Estimated values for an acute corner of $\theta_c = 60^o$.

Figure 3.15: Estimated values of range r, inclination angle θ, and angle of acute corner θ_c, without noise for acute corners of $\theta_c = 30^o$, 45^o and 60^o.



——: Estimated values for an acute corner of $\theta_c = 30^o$.

- - - : Estimated values for an acute corner of $\theta_c = 45^o$.

. . .: Estimated values for an acute corner of $\theta_c = 60^o$.

Figure 3.16: Estimated values of range r, inclination angle θ, and angle of acute corner θ_c, with $\sigma_A = 0.002$ for acute corners of $\theta_c = 30^o$, 45^o and 60^o.

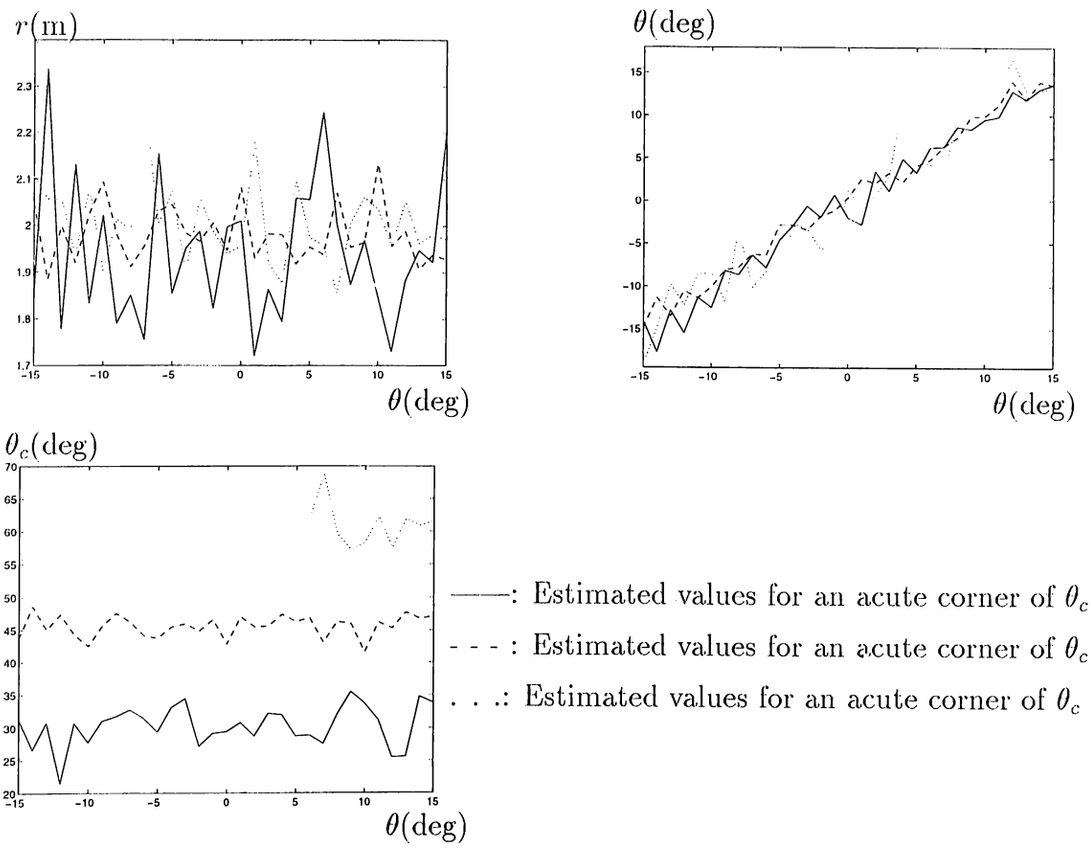


Figure 3.17: Estimated values of range r, inclination angle θ, and angle of acute corner θ_c, with $\sigma_A = 0.003$ for acute corners of $\theta_c = 30^o$, 45^o and 60^o.

	$\theta_c = 30^o$	$\theta_c = 45^o$	$\theta_c = 60^o$
$\max(t_{aa} - t_{ab})$	4.2472×10^{-4} sec	3.8316×10^{-4} sec	3.1522×10^{-4} sec
$\max(t_{bb} - t_{ab})$	4.2469×10^{-4} sec	3.8309×10^{-4} sec	3.1512×10^{-4} sec

Table 3.4: Maximum differences between TOF's for acute corners for three values of θ_c.

The maximum differences between TOF's for these acute corners are given in Table 3.4. Based on these values and the relation between σ_A and σ_t given in Table 3.3, classification results of a single transducer pair are given in Figure 3.18. Noise standard deviation of σ_A in the interval [0, 0.008] is added to the echo amplitude waveform at each angle θ and the corresponding noisy TOF's are found from this noisy amplitude waveform. In the classification algorithms, these noisy TOF and amplitude measurements with the corresponding σ_A and

σ_t values provided in Table 3.3 are used. The probability of correct classification is one and probability of misclassification and unknown are zero for all σ_A values. This result is an expected one, since maximum value of standard deviation of TOF estimates is about 8 μsec which corresponds to about 2% of maximum differences between TOF's for acute corners. Increasing σ_A to obtain higher σ_t is meaningless since $\sigma_A = 0.008$ corresponds to 100% of the maximum difference between the echo amplitudes. Hence, after noiseless TOF and amplitude measurements are extracted for all stepping angles, noise of $\sigma_t > 7.61$ μsec is added to TOF measurements while σ_A is kept constant at 0.008 and added to the amplitude measurements.

In Figure 3.18, the probability of correct classification is higher than both the probability of misclassification and the probability of unknown target up to $\sigma_t = 160$ μsec for $\theta_c = 30^o$, $\sigma_t = 140$ μsec for $\theta_c = 45^o$ and $\sigma_t = 80$ μsec for $\theta_c = 60^o$. Beyond $\sigma_t = 160$ μsec, the probability of unknown exceeds both the probability of correct classification and misclassification. For all σ_t values, the probability of misclassification is around zero due to the inclusion of the σ_t and the σ_A terms which are used in the classification algorithms. The classification result for the acute corner of $\theta_c = 60^o$ when σ_t and σ_A parameters are set to zero in the differentiation algorithm can be seen in Figure 3.19. Comparing this figure with Figure 3.18(c), the probability of misclassification is much higher for all σ_t values although the probability of correct classification has improved for $\sigma_t \geq 250$ μsec.

The extension of σ_t beyond $\sigma_t = 7.61$ μsec is not very realistic but it is highly useful for seeing the change in the classification if we have TOF noise standard deviations higher than 8 μsec in a real system.

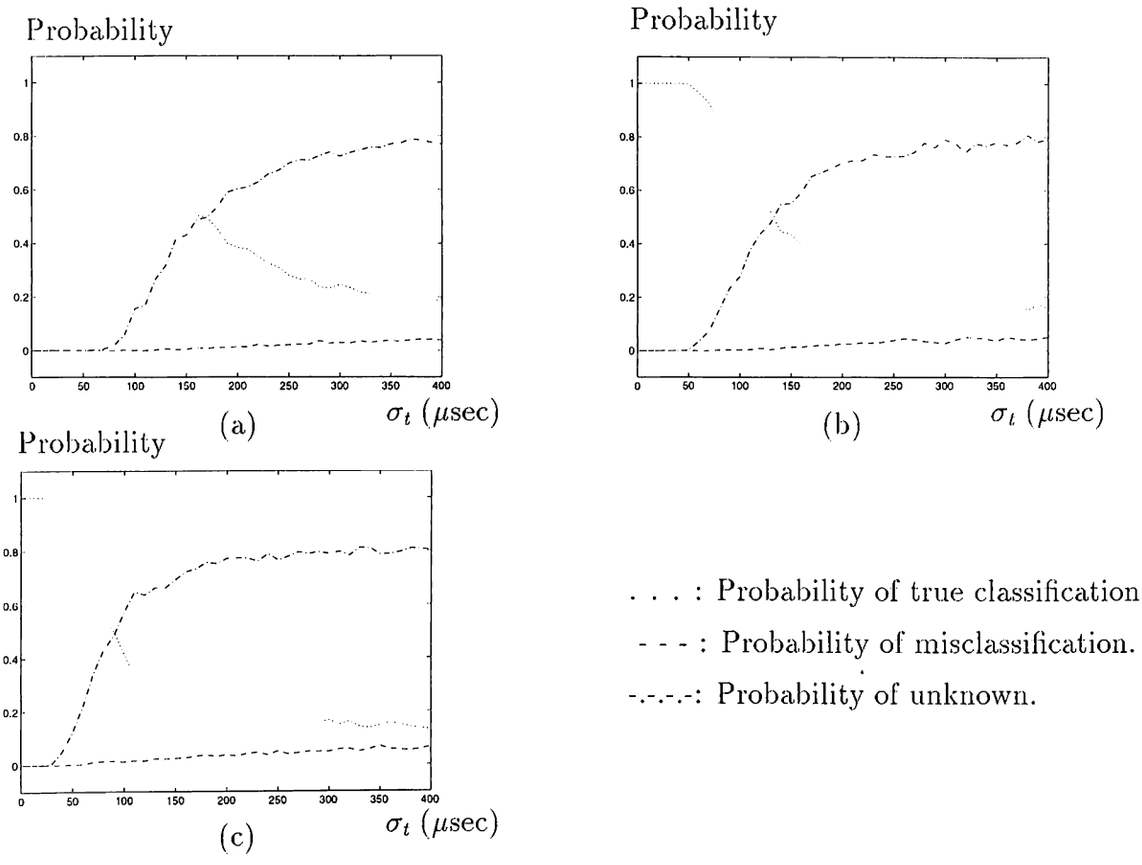


Figure 3.18: Classification results with a single transducer pair by the second approach when an acute corner at $r = 2$ m is scanned with (a) $\theta_c = 30^o$ (b) $\theta_c = 45^o$ (c) $\theta_c = 60^o$.

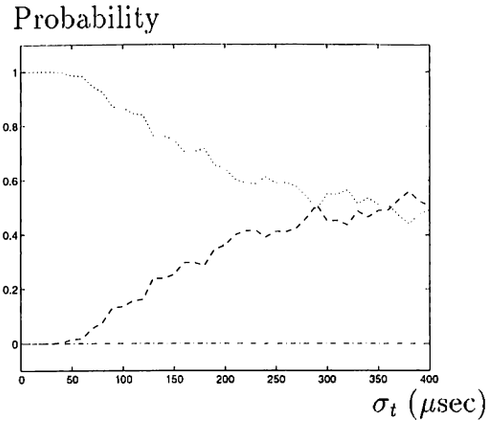


Figure 3.19: The classification result for the acute corner of $\theta_c = 60^o$ when σ_t and σ_A terms are replaced with zero in the corresponding algorithms.

In this chapter, a logical sensing module is introduced. Feature fusion with Dempster-Shafer rule of combination from multiple sonars for plane, corner

and acute corner identification is developed. Range and azimuth estimates are fused. Simulation results are also provided. Theoretical foundations are supported with these simulation results. In the next chapter, TOF and amplitude characteristics of target primitives are verified with real data.

Chapter 4

VERIFICATION WITH EXPERIMENTAL DATA

In this chapter, simulation results are verified by obtaining differentials in TOF and amplitude characteristics of the target primitives. Beliefs to plane, corner and acute corner are assigned by using these differential characteristics. Experimental results from a simple rectangular room are presented to demonstrate the fusion process using three transducer pairs.

4.1 Experimental Set-up

In this study, two different experimental set-ups are employed to investigate the TOF and amplitude characteristics of the target primitives. Both systems are mounted on a small 6 V stepper motor whose stepping angle is 0.9^o. The motion of the stepper motor is controlled by the parallel port of an IBM-PC 486 and the aid of a microswitch. Data acquisition from the sonars is accomplished by using a metrabyte DAS-50 A/D card with 12 bit resolution and 1 MHz sampling frequency. Echo signals are processed on an IBM-PC 486 using the C language.

Data is acquired sequentially to benefit maximally from the high sampling rate of the A/D card. In other words, first transducer a is excited and the echo received at the transducer below a is recorded, next a is excited and echo at the transducer below b is recorded, then b is excited and the echo at the transducer below b is sampled and recorded. Finally b is excited and the echo

at the transducer below a is recorded. Starting at the transmit time, 10,000 samples of each echo signal have been collected and thresholded. Since the amplitude noise standard deviation on the system is approximately 20 mV, the threshold level was set equal to 100 mV which corresponds to five times the noise standard deviation and 4% of the maximum amplitude range of A/D card. The amplitude information is extracted by finding the maximum value of the signal after the threshold value is exceeded. The targets employed in this study are: cylinders with radii 1.5 cm, 2.5 cm, 5.0 cm, 7.5 cm, a planar target, a corner and an acute corner of $\theta_c = 60^o$. The cylindrical targets with radii 1.5 cm and 2.5 cm are considered good approximations to an edge type target since cylinders with small radii behave similarly to outer edges formed by the intersection of two planes [1].

All of the experiments are conducted on large sheets of milimetric paper to allow accurate calibration. In the experiments, each target's surface distance r to the center of the transducer system is varied between 20 to 140 cm at 10 or 20 cm intervals. At each position, data is collected while the target is stationary at $\theta = 0^o$. The typical differential TOF between the transducers varies between 0−14 cm depending on the target type, curvature and distance for fixed separations of $d = 8$ cm, $d = 15$ cm and $d = 24$ cm. As the range of the target increases, the differential signal becomes less reliable for target classification.

4.2 Experimental Results with Polaroid Transducers

In the first system, two Polaroid transducers are used with separation $d = 15$ cm as illustrated in Figure 4.2. The aperture radius of the transducers is $a = 2.0$ cm, and the resonant frequency is $f_o = 49.4$ kHz. Therefore $\theta_o \cong 12^o$ for these transducers.

TOF and amplitude characteristics of cylinders whose radii are 2.5, 5.0, 7.5 cm and TOF characteristics of a plane at various ranges can be obtained with this system. Differentials in these characteristics and the theoretical predictions at specific ranges are given in Figures 4.1, 4.3−4.9. In this system, TOF characteristics of plane are taken at every 10 cm for the ranges between 30 cm and 140 cm. However, TOF and amplitude characteristics of cylinders can be taken at every 10 cm for the ranges between 50 cm and 140 cm. These

characteristics are compared with the theoretical results at the range value $r = 80$ cm. For the cylinders, experimental TOF characteristics closely follow the theoretical ones when $-6^o \leq \theta \leq 6^o$. Experimentally obtained amplitude characteristics have almost the same shape as the theoretical ones (Figures 4.4, 4.6 and 4.8). Note that, the decrease in amplitude when range increases is not seen in these figures due to the range-time-gain amplifier in the Polaroid system. Moreover, obtained TOF characteristics for the planar target closely follow the theoretical predictions for $-15^o \leq \theta \leq 15^o$ in this system (Figure 4.1).

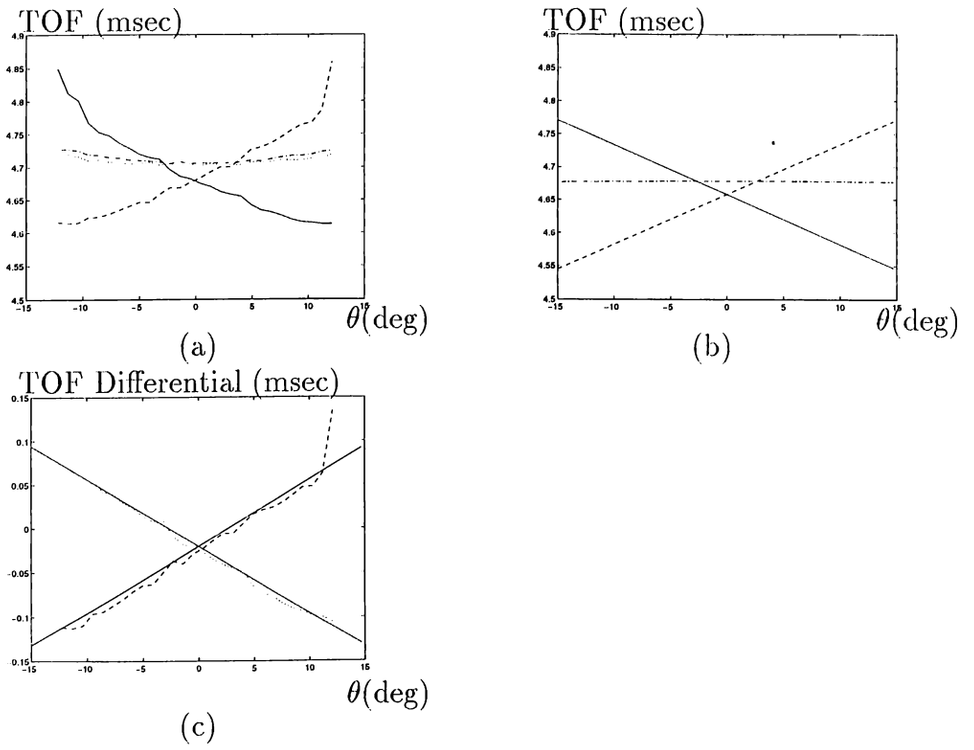


Figure 4.1: Comparison of TOF characteristics of a plane and the theoretical predictions when a Polaroid transducer pair with separation $d = 15$ cm is employed at $r = 80$ cm (a) experimental results (b) theoretical results (c) differential TOF.

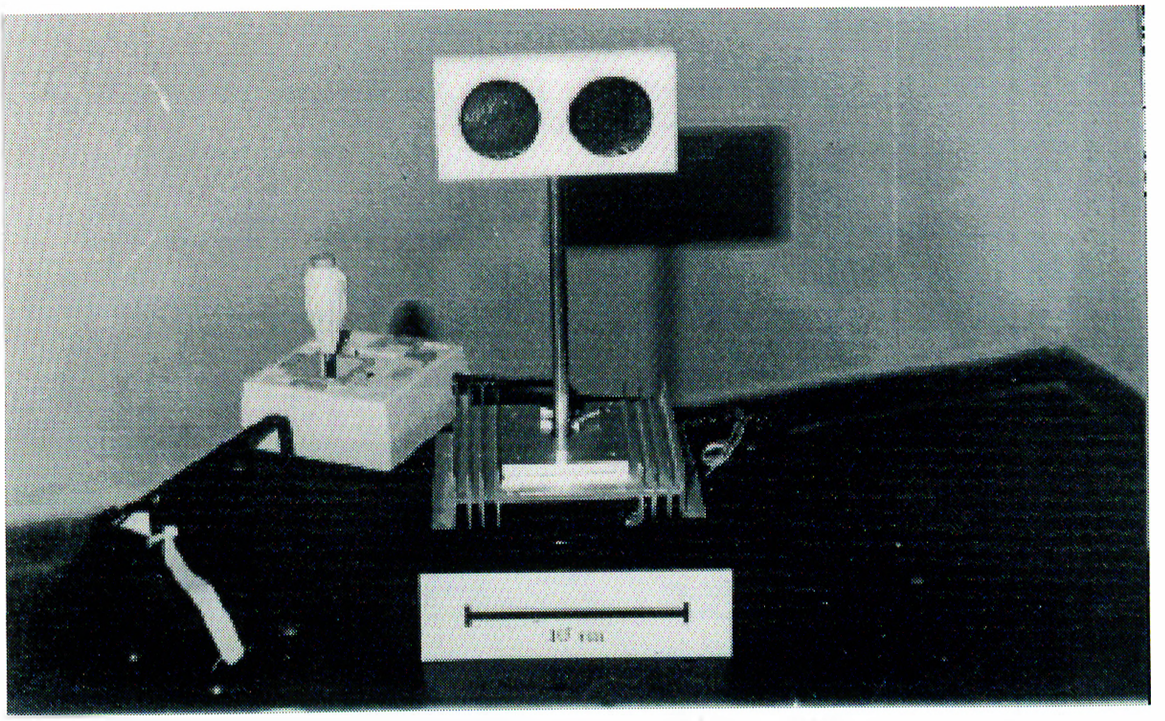


Figure 4.2: Configuration of the Polaroid transducer pair in the real system.

In this figure, the illustrated center-to-center transducer separation is 5.0 cm.

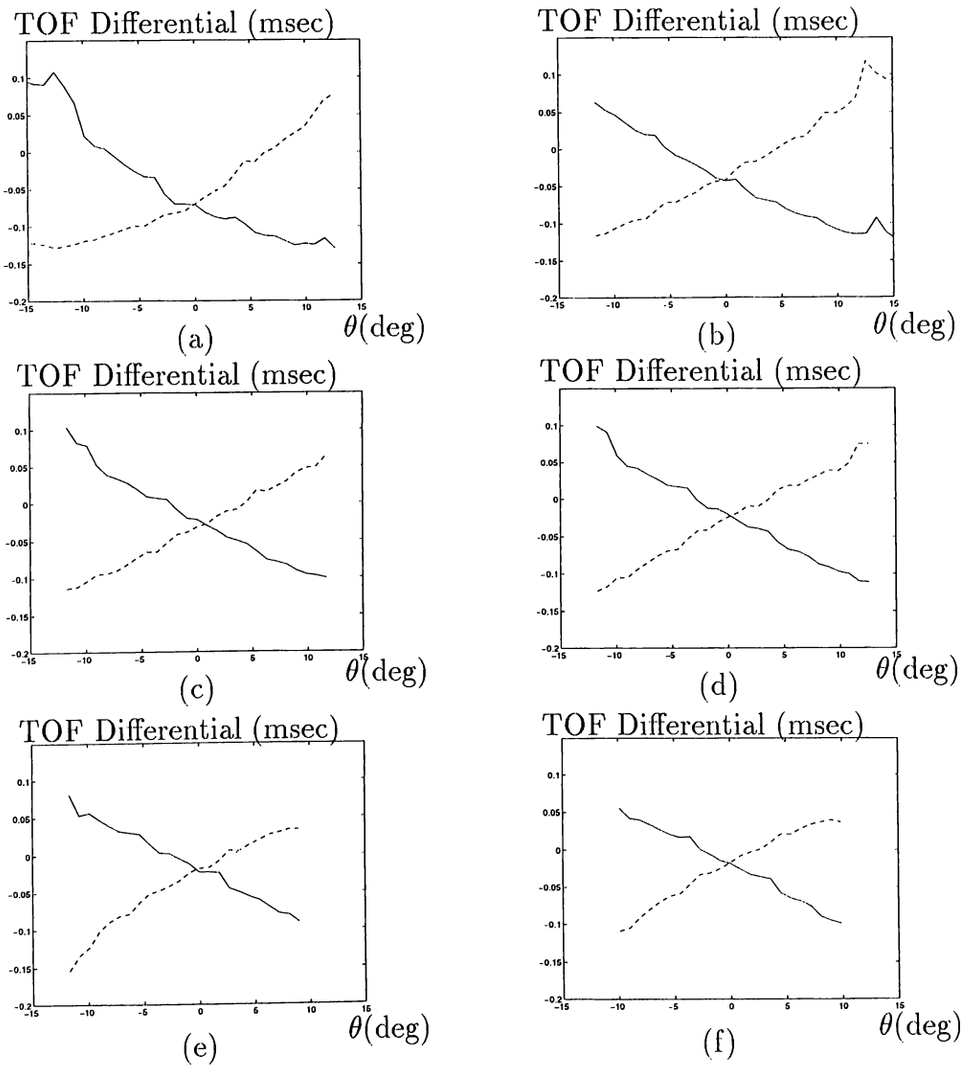


Figure 4.3: Differentials in TOF characteristics of a plane when a Polaroid transducer pair with separation $d = 15$ cm is employed at the range values (a) r =30 cm (b) r =50 cm (c) r =70 cm (d) r =90 cm (e) r =110 cm (f) r =130 cm.

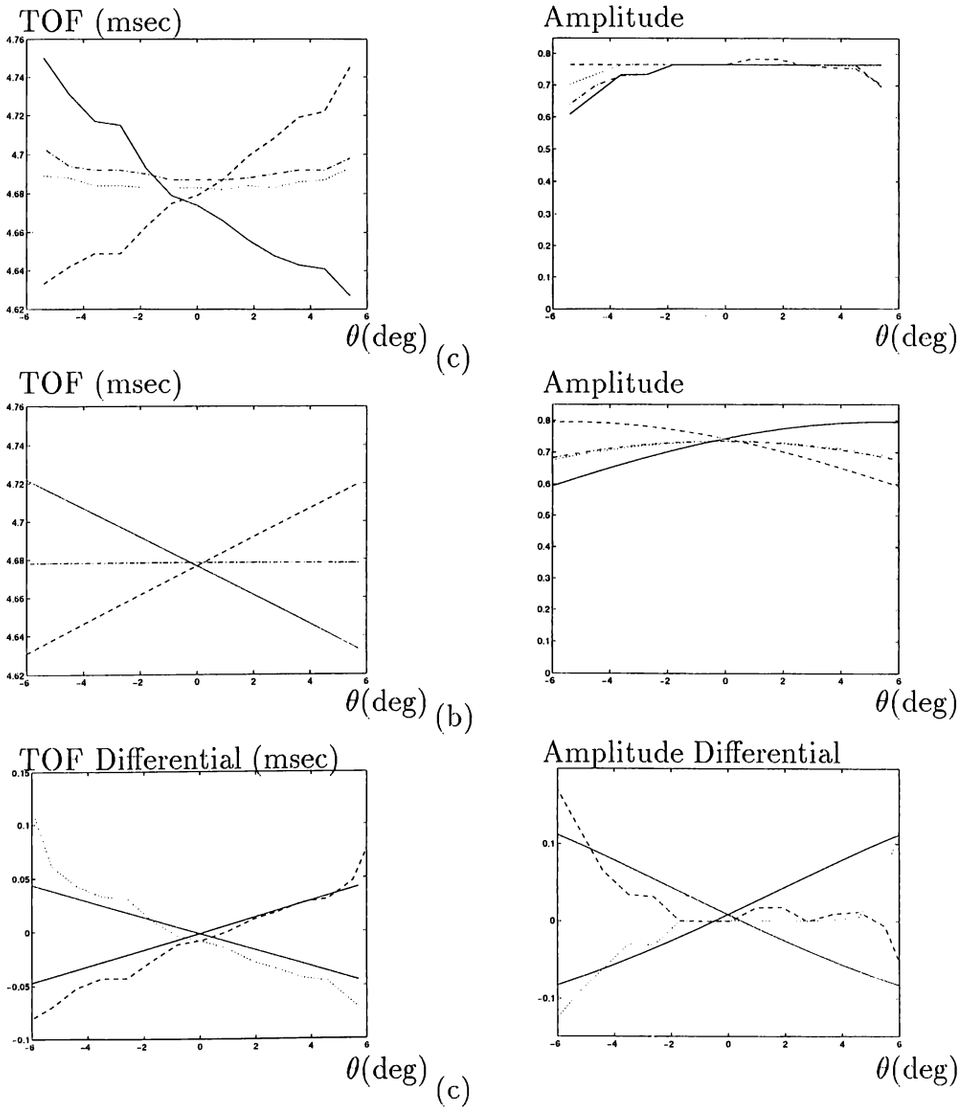


Figure 4.4: Comparison of TOF and amplitude characteristics of a cylinder of $r_c = 7.5$ cm with the theoretical predictions when a Polaroid transducer pair with separation $d = 15$ cm is employed at $r = 80$ cm (a) experimental results (b) theoretical results (c) differential signals.

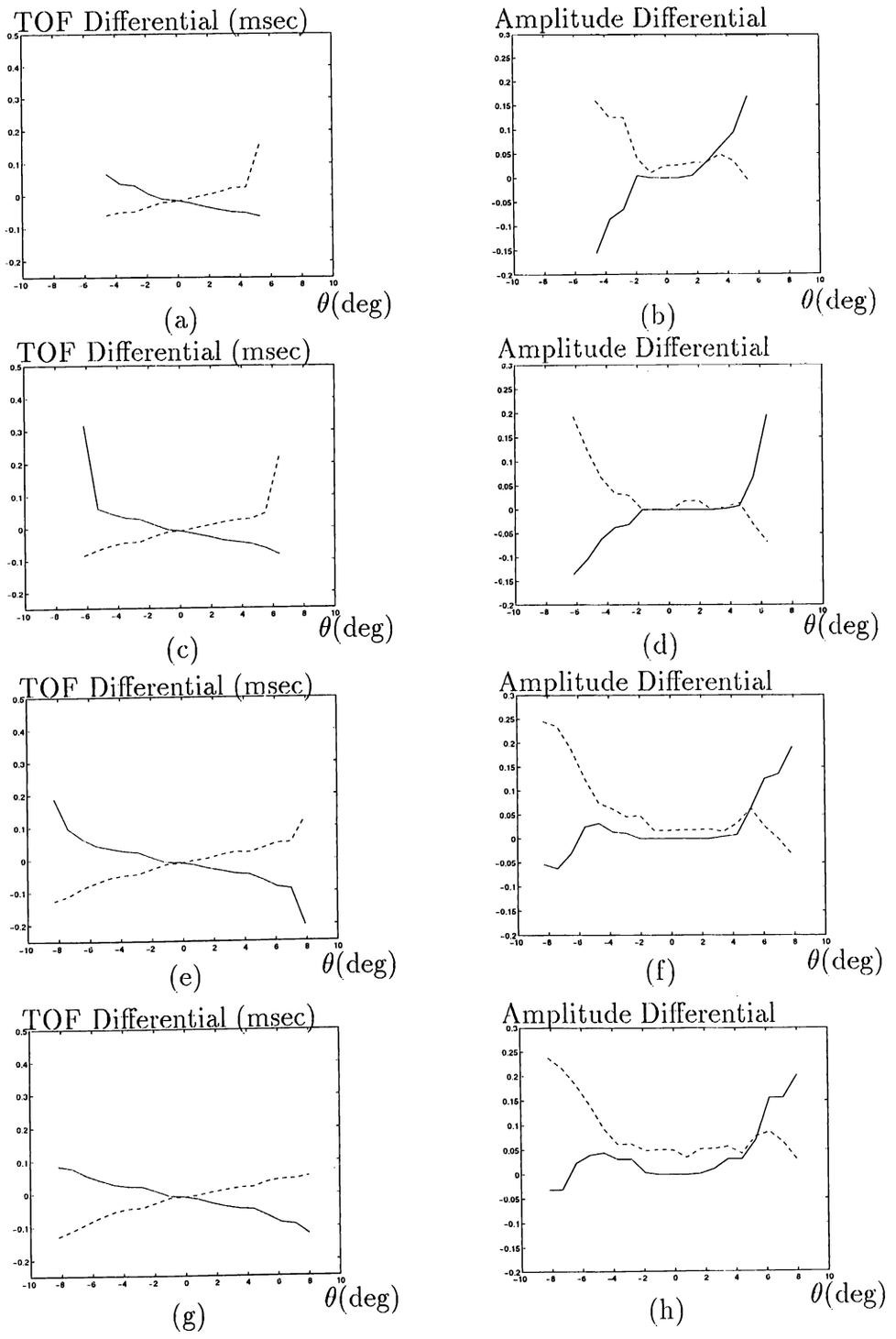


Figure 4.5: Differentials in TOF and amplitude characteristics cylinder with $r_c r_c$ = 7.5 cm when a Polaroid transducer pair with separation d = 15 cm is employed at the range values (a),(b) r =50 cm (c),(d) r =70 cm (e),(f) r =90 cm (g),(h) r =110 cm.

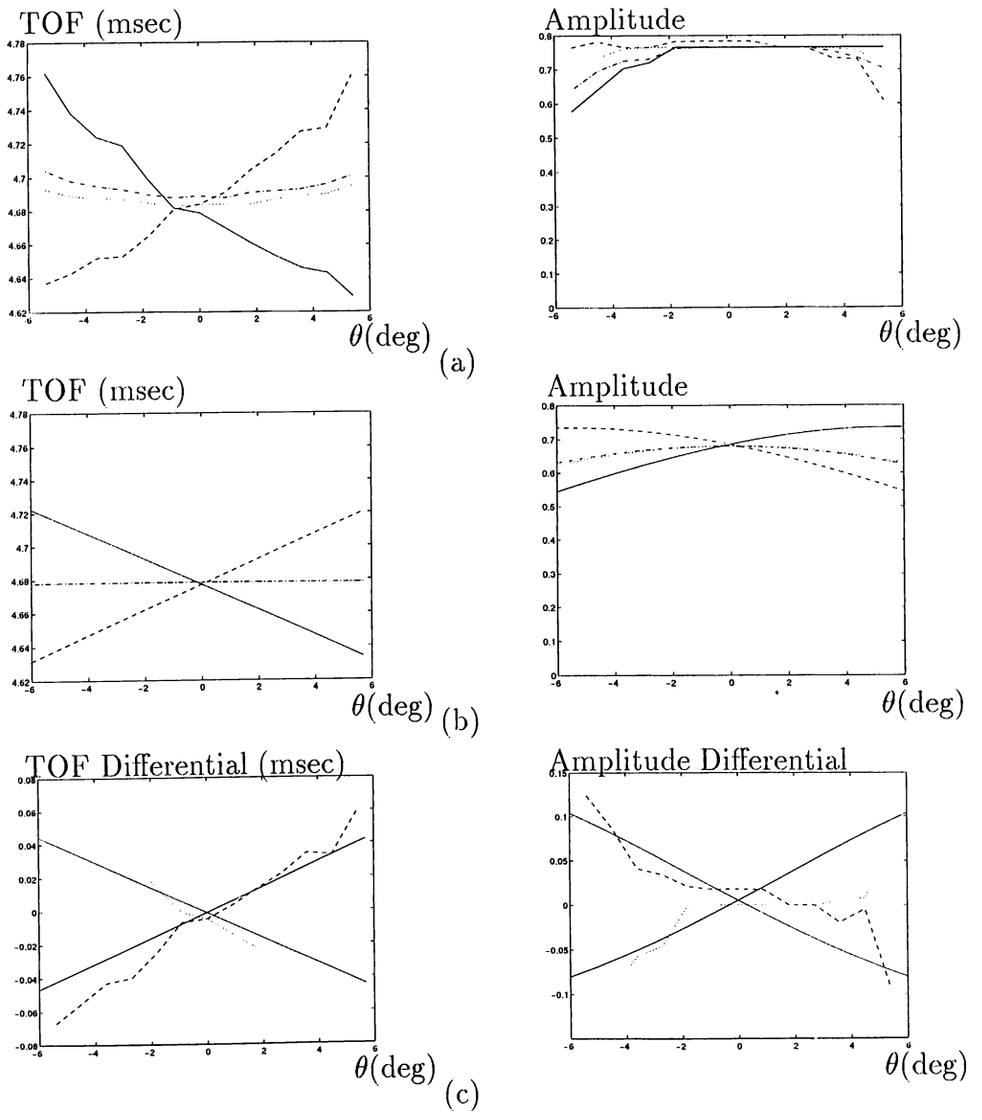


Figure 4.6: Comparison of TOF and amplitude characteristics of a cylinder of $r_c = 5.0$ cm with the theoretical predictions when a Polaroid transducer pair with separation $d = 15$ cm is employed at $r = 80$ cm (a) experimental results (b) theoretical results (c) differential signals.

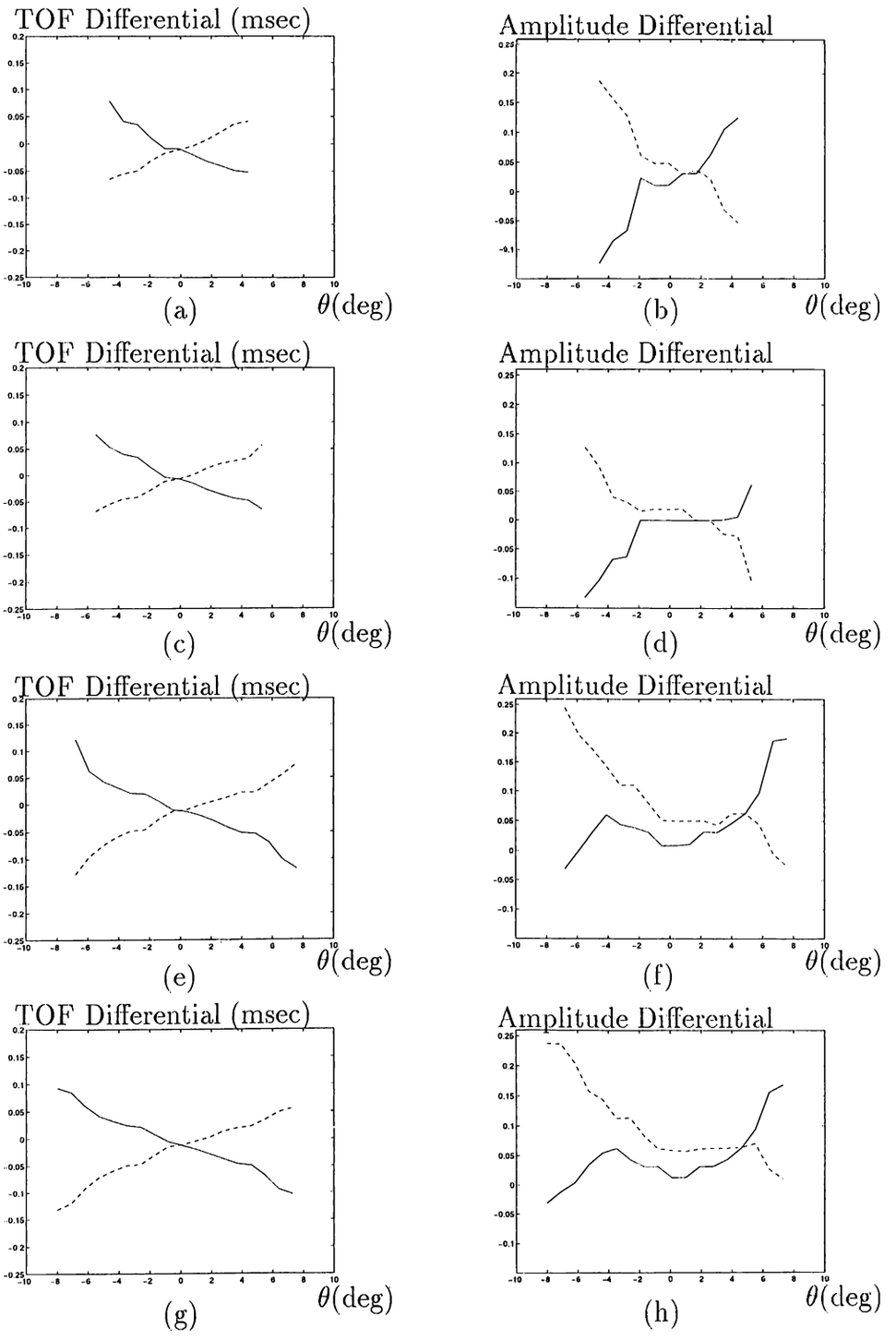


Figure 4.7: Differentials in TOF and amplitude characteristics of a cylinder with $r_c = 5.0$ cm when a Polaroid transducer pair with separation $d = 15$ cm is employed at the range values (a),(b) r =50 cm (c),(d) r =70 cm (e),(f) r =90 cm (g),(h) r =110 cm.

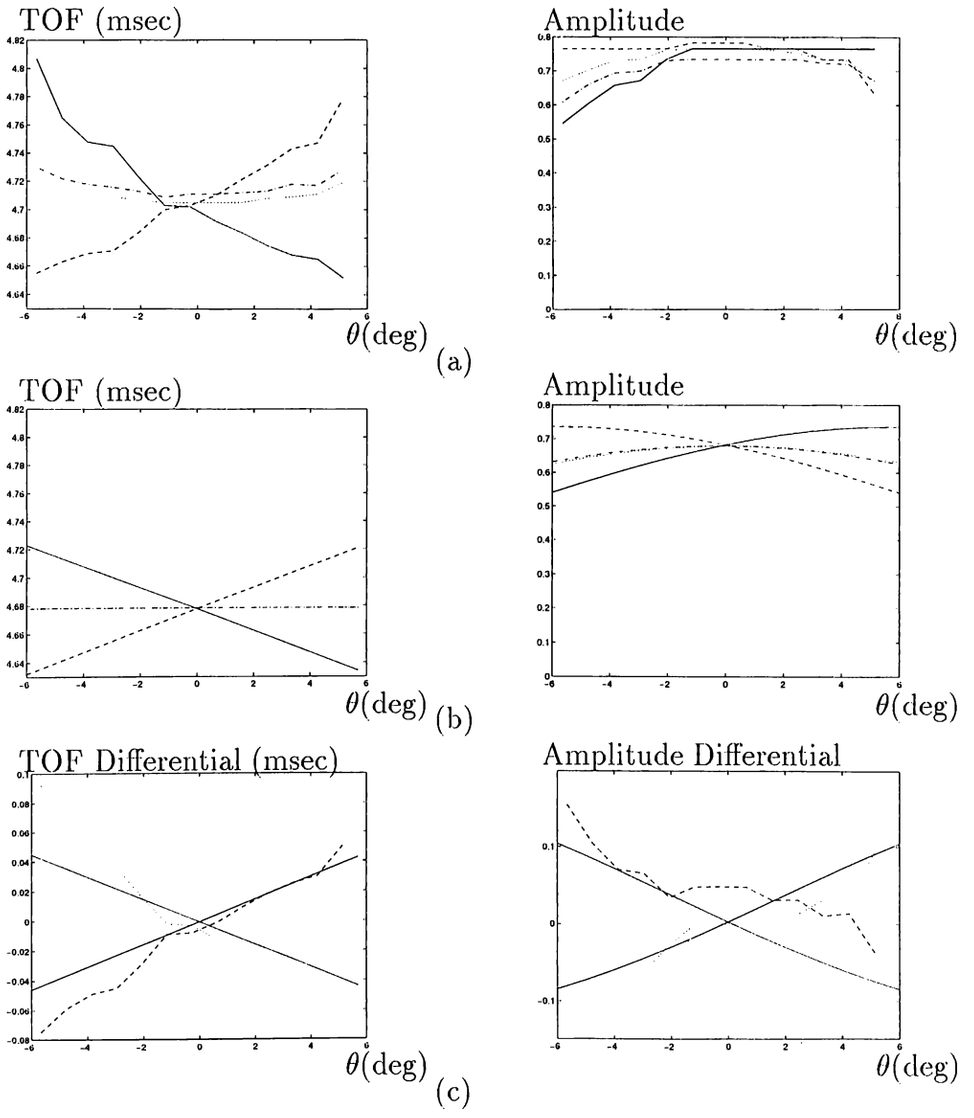


Figure 4.8: Comparison of TOF and amplitude characteristics of a cylinder of $r_c = 2.5$ cm with the theoretical predictions when a Polaroid transducer pair with separation $d = 15$ cm is employed at $r = 80$ cm (a) experimental results (b) theoretical results (c) differential signals.

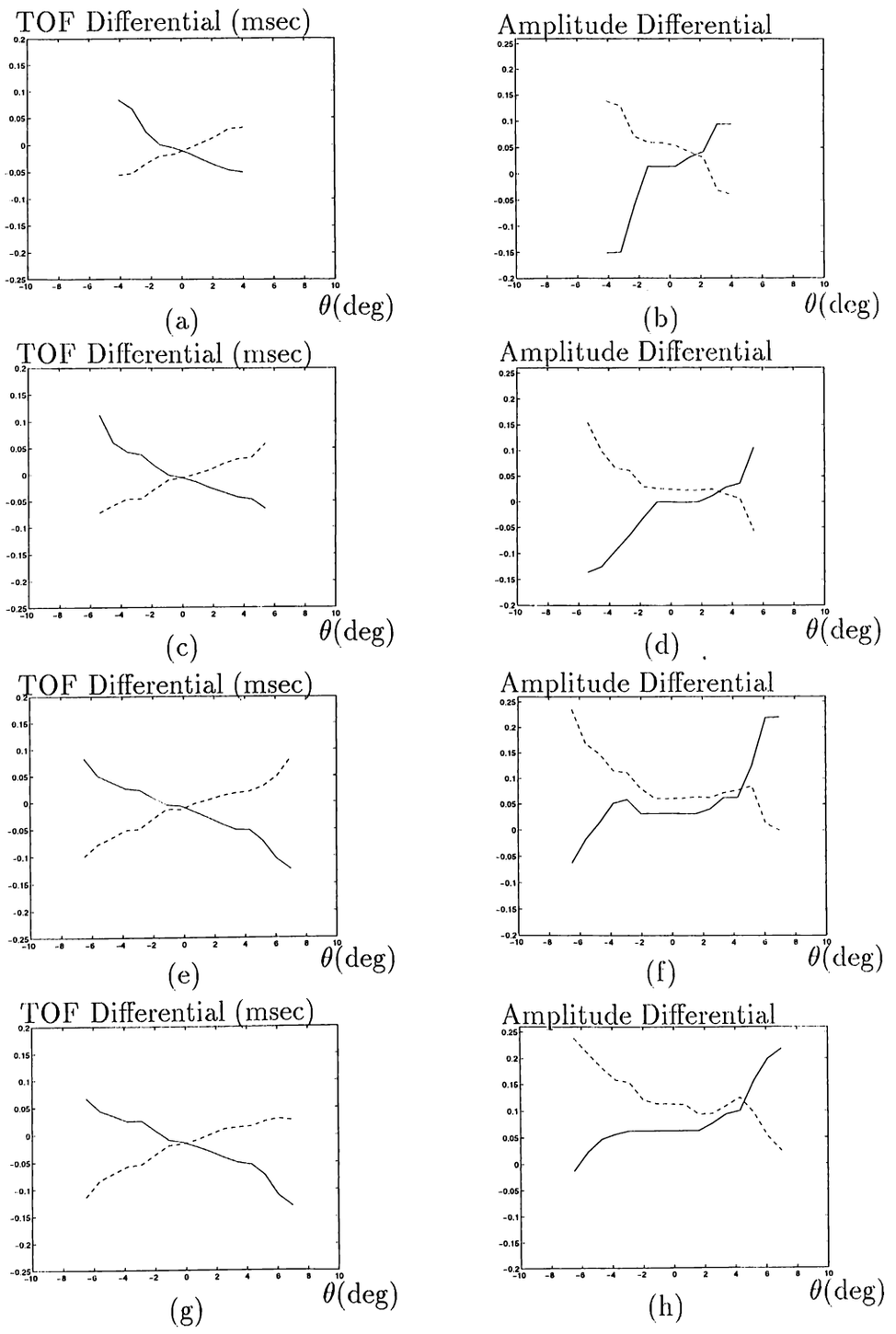


Figure 4.9: Differentials in TOF and amplitude characteristics of a cylinder with $r_c = 2.5$ cm when a Polaroid transducer pair with separation $d = 15$ cm is employed at the range values (a),(b) r =50 cm (c),(d) r =70 cm (e),(f)r =90 cm (g),(h) r =110 cm.

4.3 Experimental Results with Panasonic Transducers

In the second system, Panasonic transducers are used which have much wider beam-width than the Polaroid transducers. Since these transducers are manufactured with different characteristics for transmitting and receiving, we have used two separate transducers for transmission and reception of the echoes. Hence, a pair of very closely spaced transmitter and receiver approximately correspond to a single transducer in the Polaroid system (Figure 4.10). Internal construction of a Panasonic transducer is illustrated in Figure 4.11. The aperture radius of the Panasonic transducer is $a = 0.65$ cm and resonant frequency is $f_o = 40$ kHz, therefore $\theta_o \cong 54^o$ for these transducers. In this system, gain is constant unlike the Polaroid system.

TOF and amplitude characteristics of plane, corner, acute corner of $\theta_c = 60^o$ and cylinders whose radii are 2.5, 5.0 and 7.5 cm are obtained at various range values. These range values are extended from 20 cm to 120 cm, but minimum and maximum range values at which data can be collected are different for different target types. These values are given in Table 4.1.

Target type	r_{min} (cm)	r_{max} (cm)
plane	20	120
corner	40	120
acute corner of $\theta_c = 60^o$	40	100
cylinder of $r_c = 2.5$ cm	40	60
cylinder of $r_c = 5.0$ cm	20	80
cylinder of $r_c = 7.5$ cm	20	80

Table 4.1: Minimum and maximum range values at which data can be collected for the target primitives.

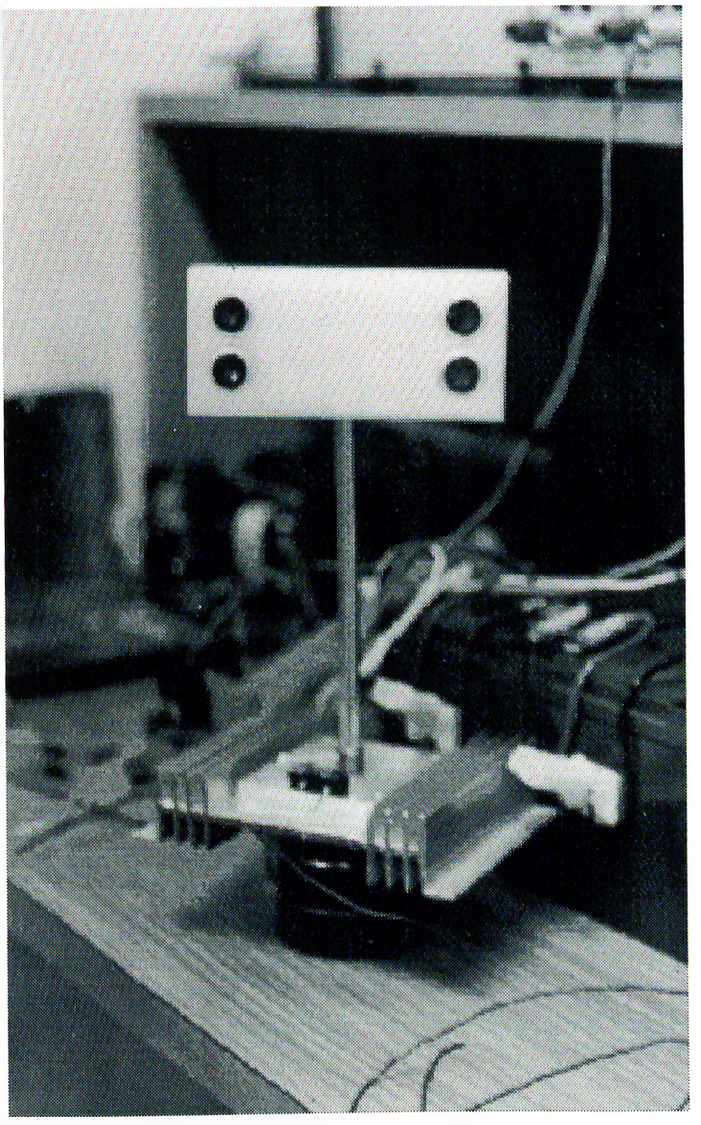


Figure 4.10: Configuration of the Panasonic transducer pair in the real system.

In this figure, the illustrated separation d is 8 cm.

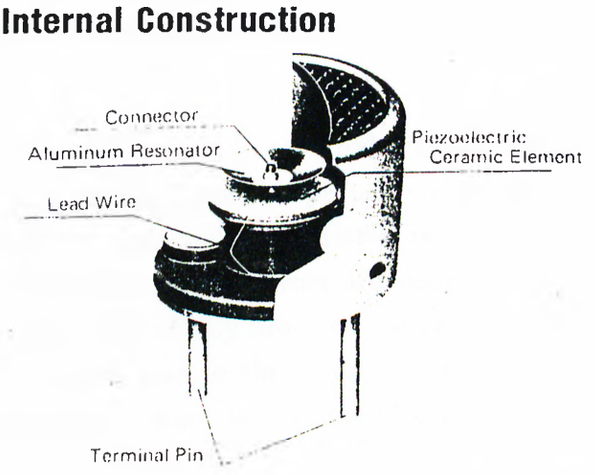


Figure 4.11: Cross-section of the Panasonic transducer.

Experimentally obtained differentials in TOF and amplitude characteristics of target primitives at various range values with this system and theoretical predictions at a specific range value are given in Figures 4.12–4.24. Since the gain in this system is not adaptive, the decrease in amplitude with increasing range can easily be observed in these figures. For planar target type, differentials in TOF and amplitude characteristics follow the theoretical results quite closely. The amplitude differentials are always greater than zero as needed for the correct differentiation of the planar target (Figures 4.13 and 4.14).

For a corner, TOF and amplitude characteristics and differentials in these characteristics follow the theoretical predictions with some error for $r \geq 40$ cm. For smaller range values, sharp decreases and increases in TOF differentials are observed since echo produced from the two planes which form the corner are detected initially.

For an acute corner of $\theta_c = 60^o$, the desired characteristics in TOF differentials is to be greater than zero since this property is used in the differentiation of target primitives. This desired characteristics is obtained only at $r = 40$ cm since the planes which form an acute corner are assumed to be of infinite length when the acute corner is physically modeled. When r is greater than 40 cm, this assumption is violated due to the physical limitations in the laboratory and desired characteristics cannot be obtained for greater range values.

For the cylindrical targets, differentials in TOF and amplitude characteristics cannot follow theoretical results as well as in the previous case. Echo amplitude which is reflected from these cylinders is small and cannot be detected very well since a linear time-gain amplifier is not used in this system. Also, for the transducer separation of $d = 24$ cm, cylindrical targets between $30-50$ cm may not be detected by the sensor pair since these targets do not fall within the sensitivity pattern of the sensor pair. However, when the transducer separation is decreased to 8 cm, in addition to these cylinder characteristics, differentials in TOF and amplitude characteristics of a cylinder with $r_c = 1.5$ cm can also be detected. These characteristics at various range values and theoretical results at a specific range are given in Figures 4.25–4.33. Referring to these figures, differentials in TOF and amplitude characteristics follow the theoretical results closely, but TOF and amplitude characteristics follow the theory with some more error.

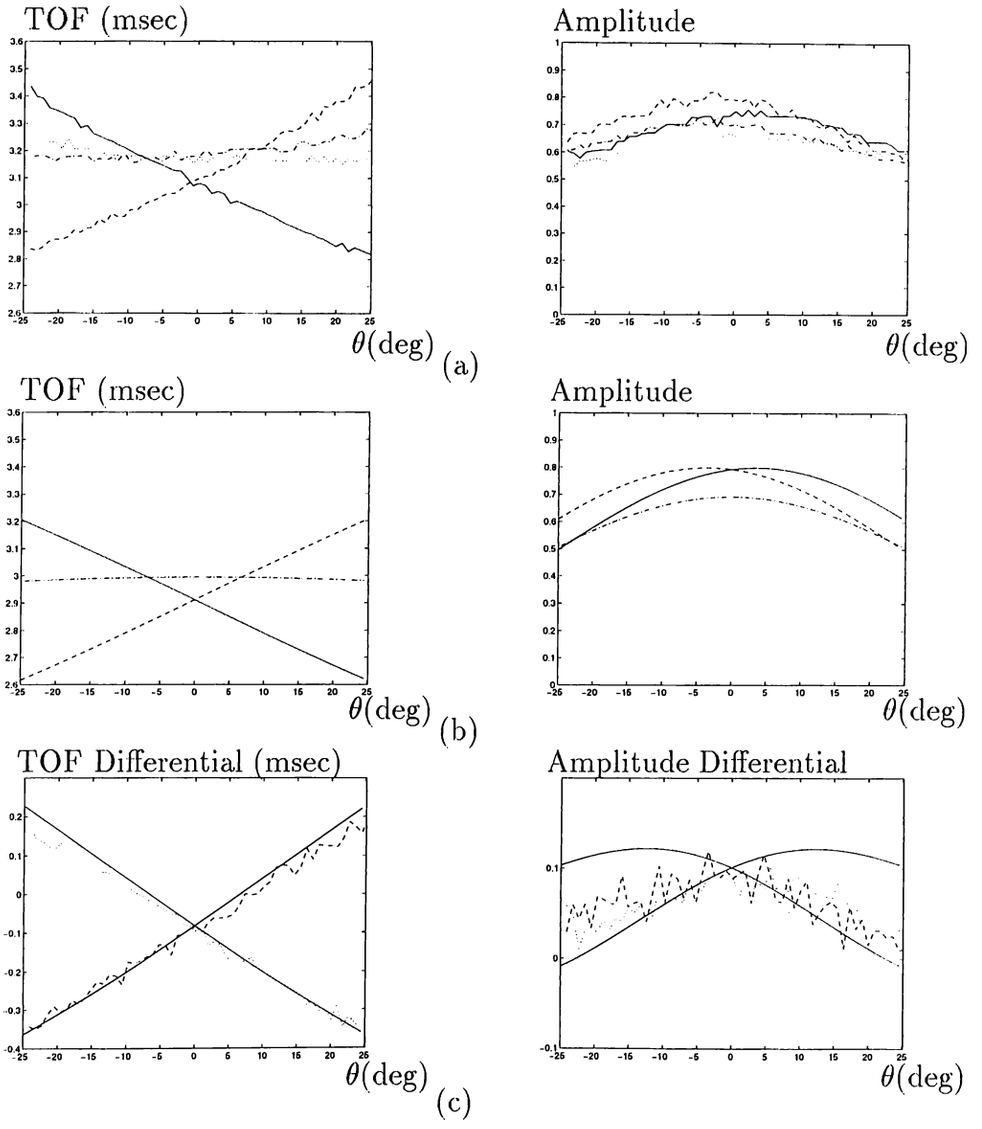


Figure 4.12: Comparison of TOF and amplitude characteristics of a plane with the theoretical predictions when a Panasonic transducer pair with separation $d = 24$ cm is employed at $r = 50$ cm (a) experimental results (b) theoretical results (c) differential signals.

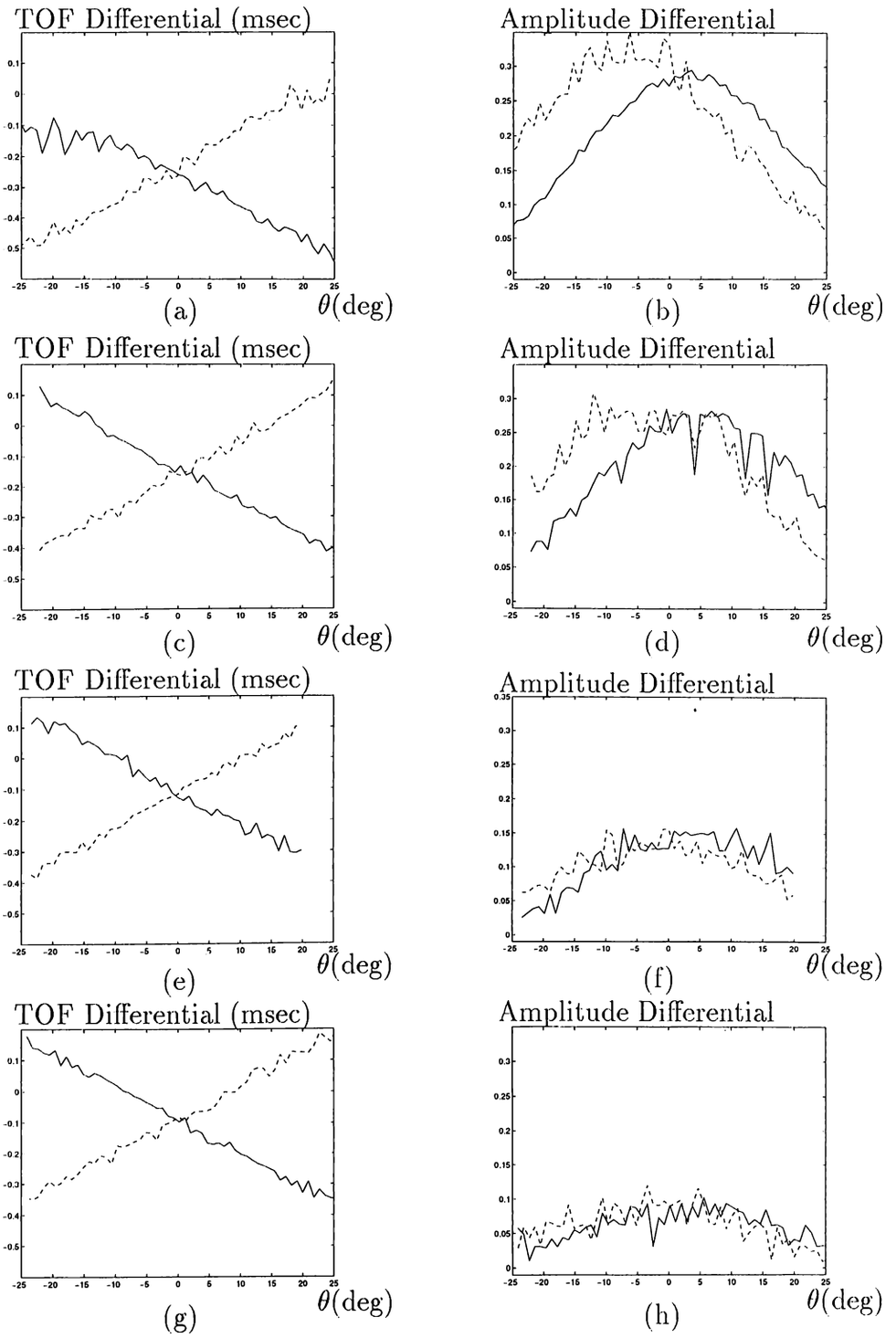


Figure 4.13: Differentials in TOF and amplitude characteristics of a plane when a Panasonic transducer pair with separation d = 24 cm is employed at the range values (a),(b) r =20 cm (c),(d) r =30 cm (e),(f) r =40 cm (g),(h) r =50 cm.

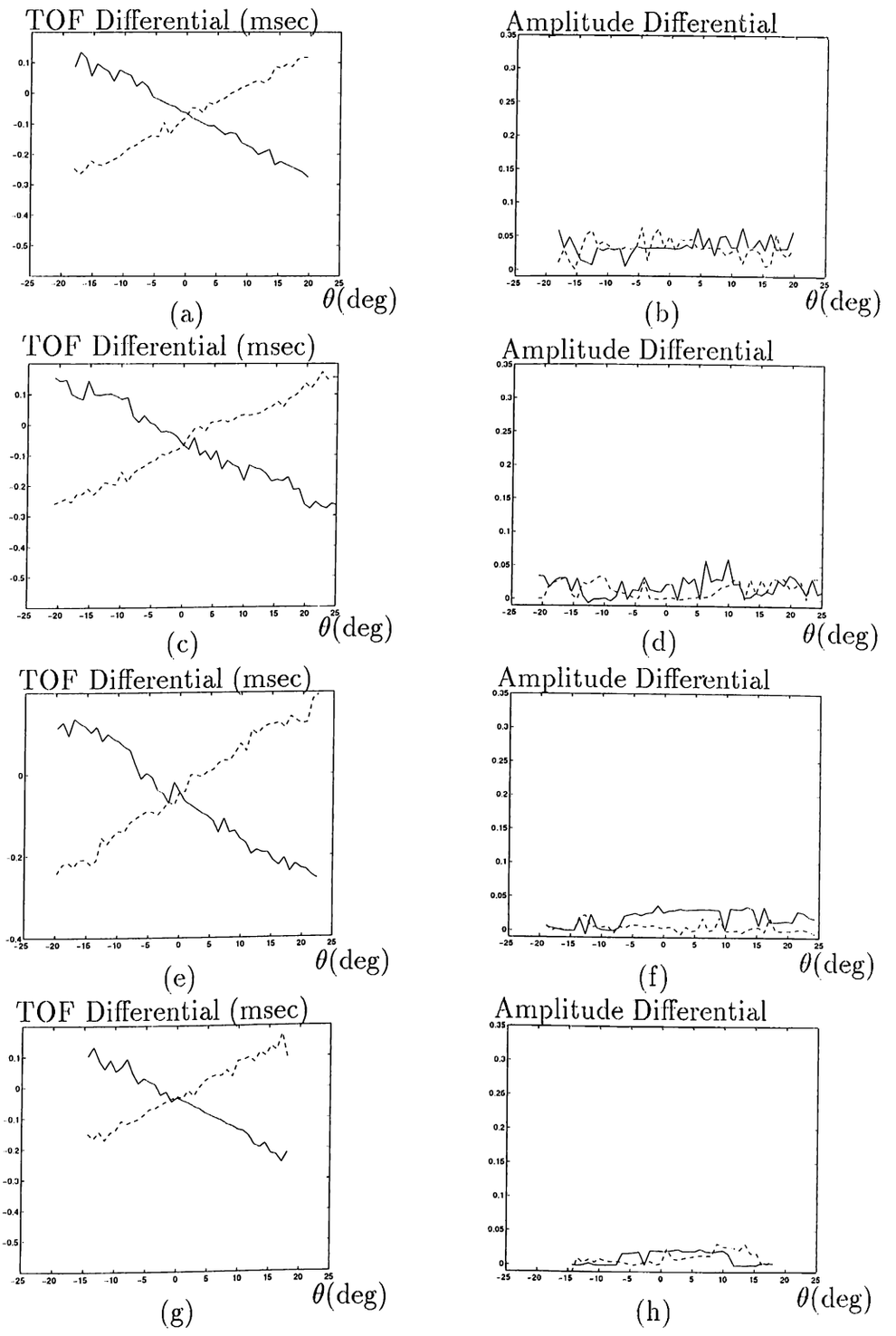


Figure 4.14: Differentials in TOF and amplitude characteristics of a plane when a Panasonic transducer pair with separation $d = 24$ cm is employed at the range values (a),(b) r =60 cm (c),(d) r =80 cm (e),(f) r =100 cm (g),(h) r =120 cm.

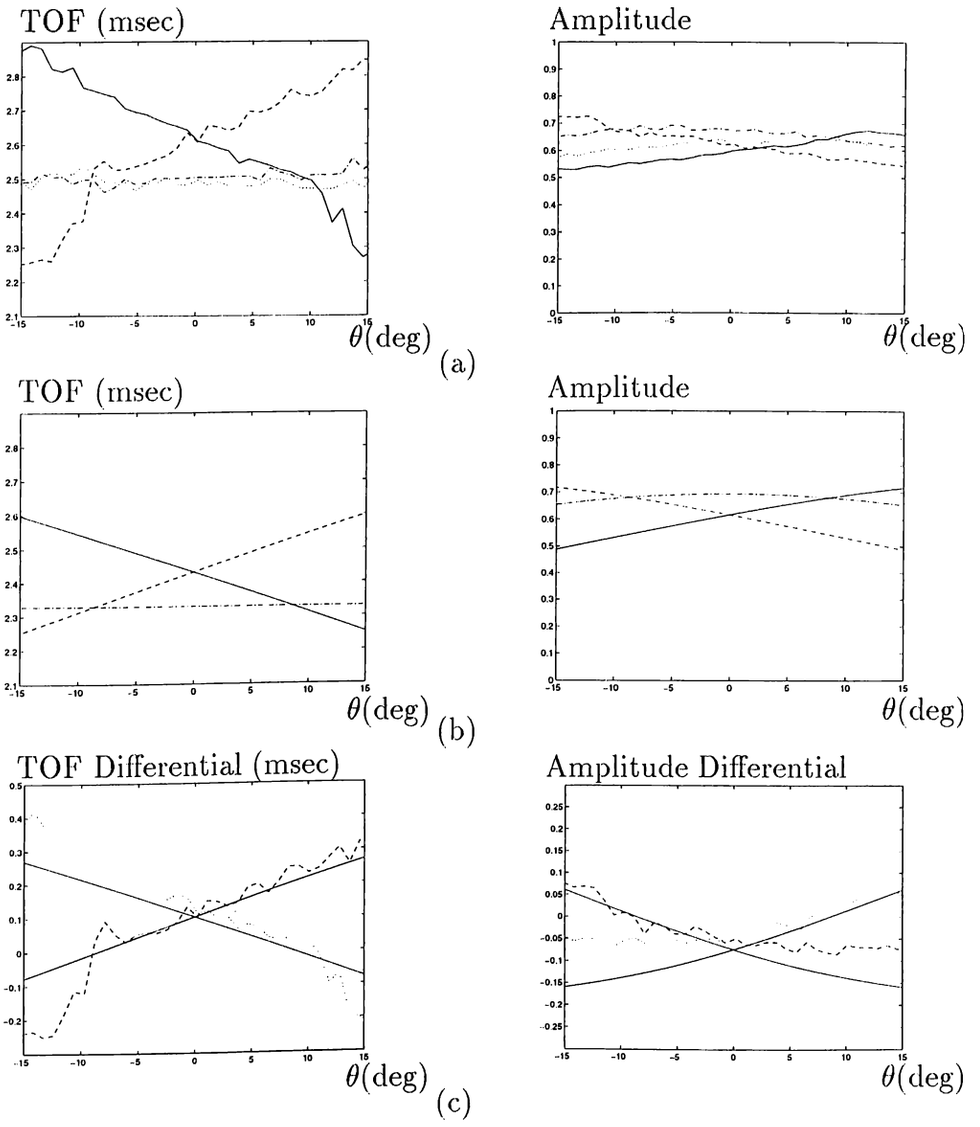


Figure 4.15: Comparison of TOF and amplitude characteristics of a corner with the theoretical predictions when a Panasonic transducer pair with separation $d = 24$ cm is employed at $r = 40$ cm (a) experimental results (b) theoretical results (c) differential signals.

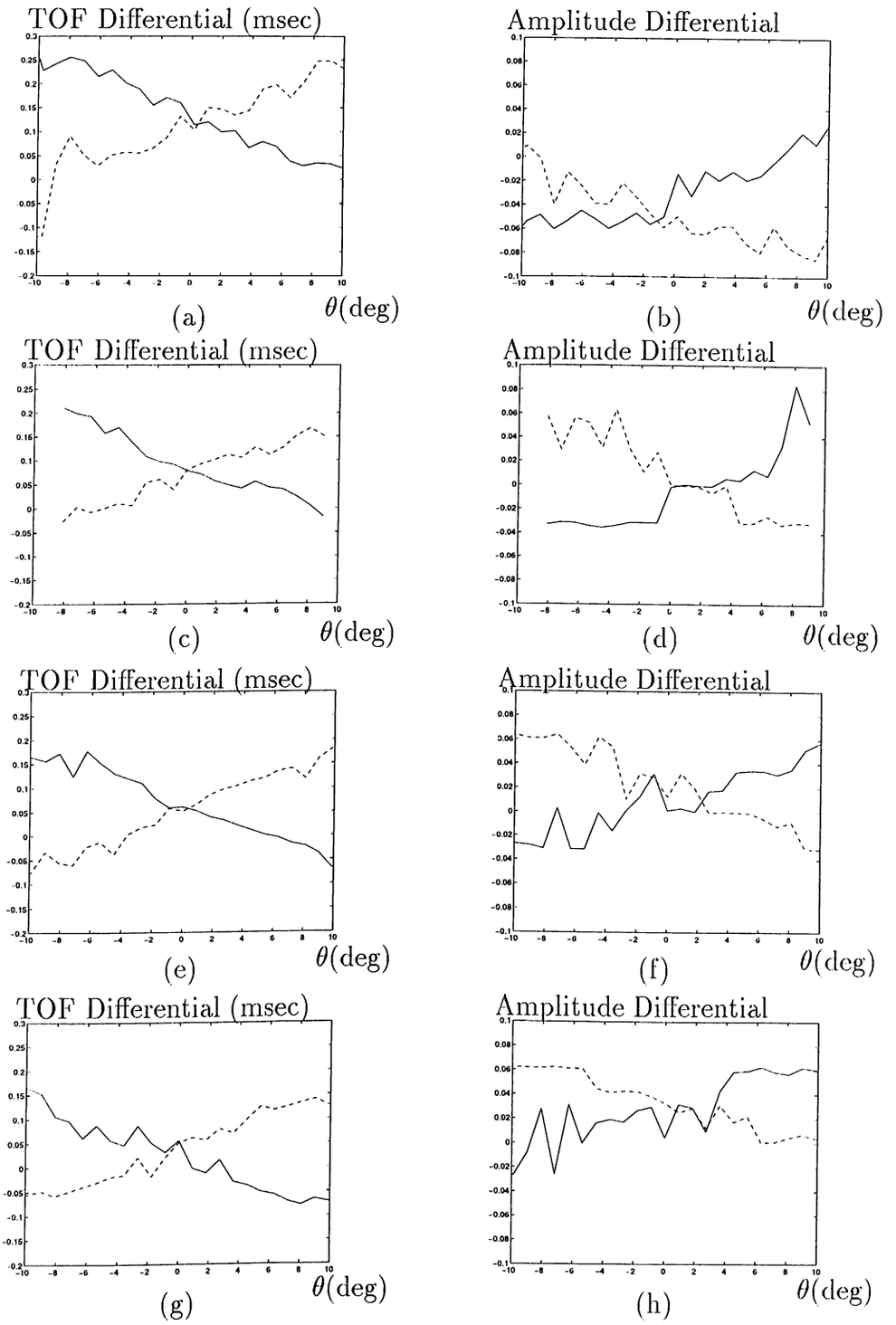


Figure 4.16: Differentials in TOF and amplitude characteristics of a corner when a Panasonic transducer pair with separation $d = 24$ cm is employed at the range values (a),(b) r-=40 cm (c),(d) r =60 cm (e),(f) r =80 cm (g),(h) r =100 cm.

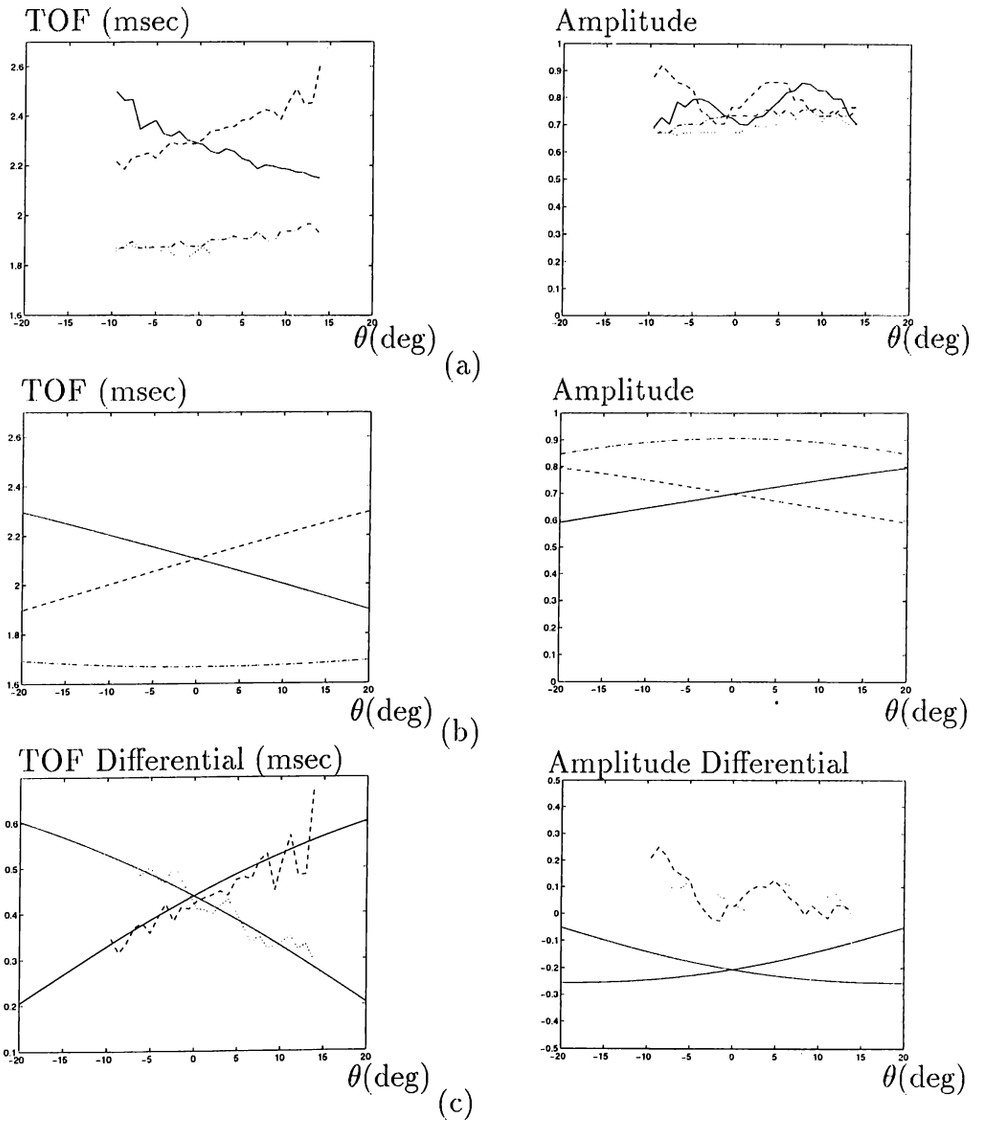


Figure 4.17: Comparison of TOF and amplitude characteristics of an acute corner of $\theta_c = 60^o$ with the theoretical predictions when a Panasonic transducer pair with separation $d = 24$ cm is employed at $r = 40$ cm (a) experimental results (b) theoretical results (c) differential signals.

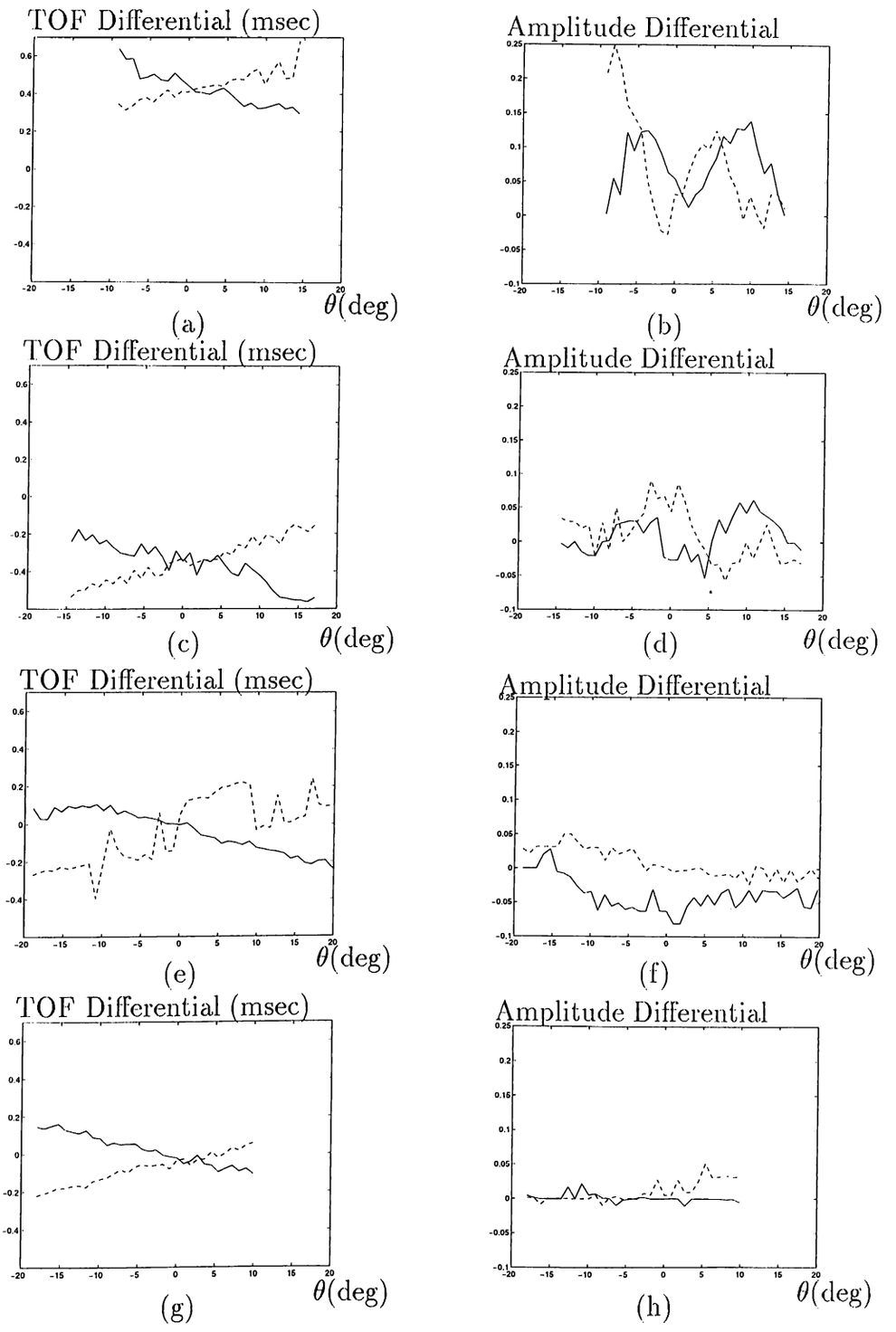


Figure 4.18: Differentials in TOF and amplitude characteristics of an acute corner of $\theta_c = 60^o$ when a Panasonic transducer pair with separation $d = 24$ cm is employed at the range values (a),(b) $r = 40$ cm (c),(d) $r = 60$ cm (e),(f) $r = 80$ cm (g),(h) $r = 100$ cm.

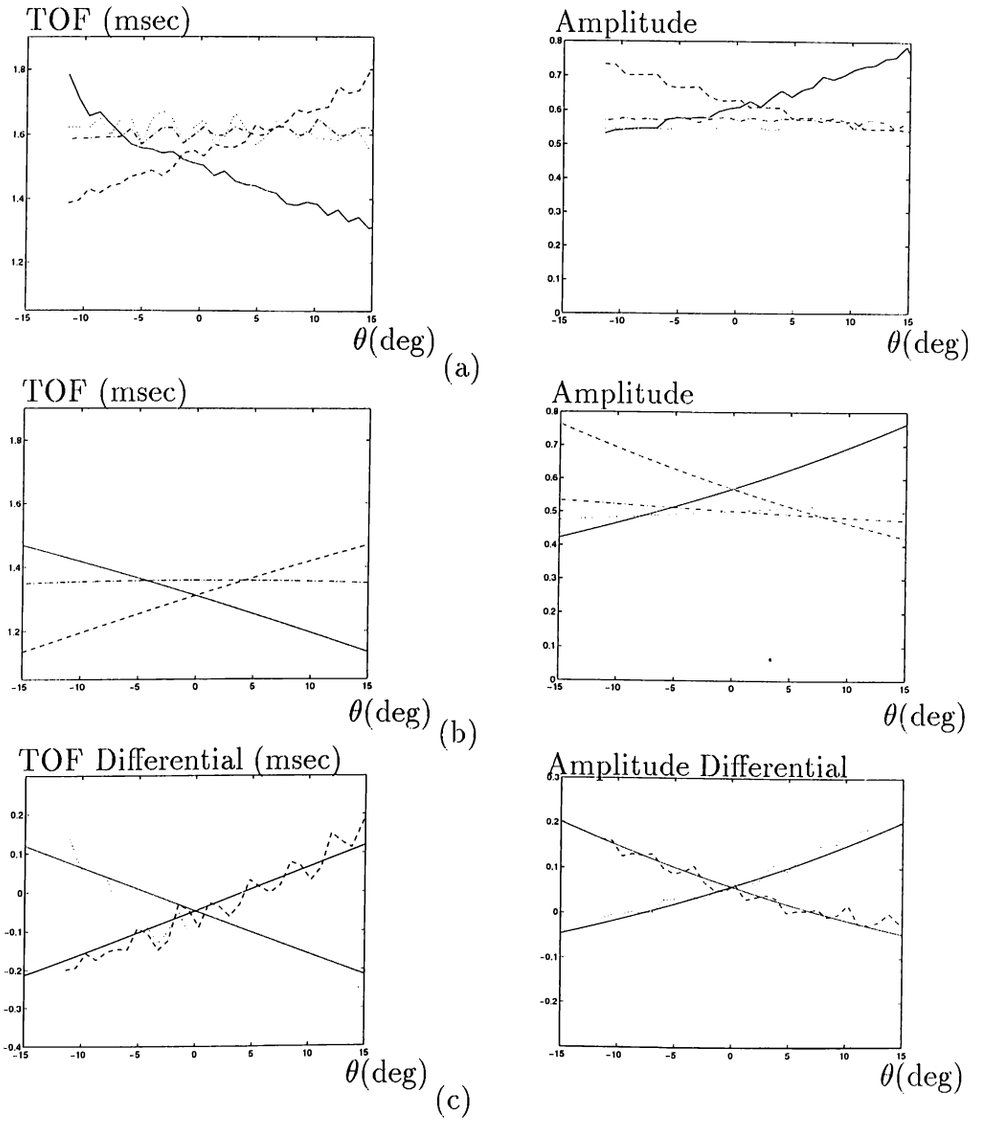


Figure 4.19: Comparison of TOF and amplitude characteristics of a cylinder of $r_c = 7.5$ cm with the theoretical predictions when a Panasonic transducer pair with separation $d = 24$ cm is employed at $r = 20$ cm (a) experimental results (b) theoretical results (c) differential signals.

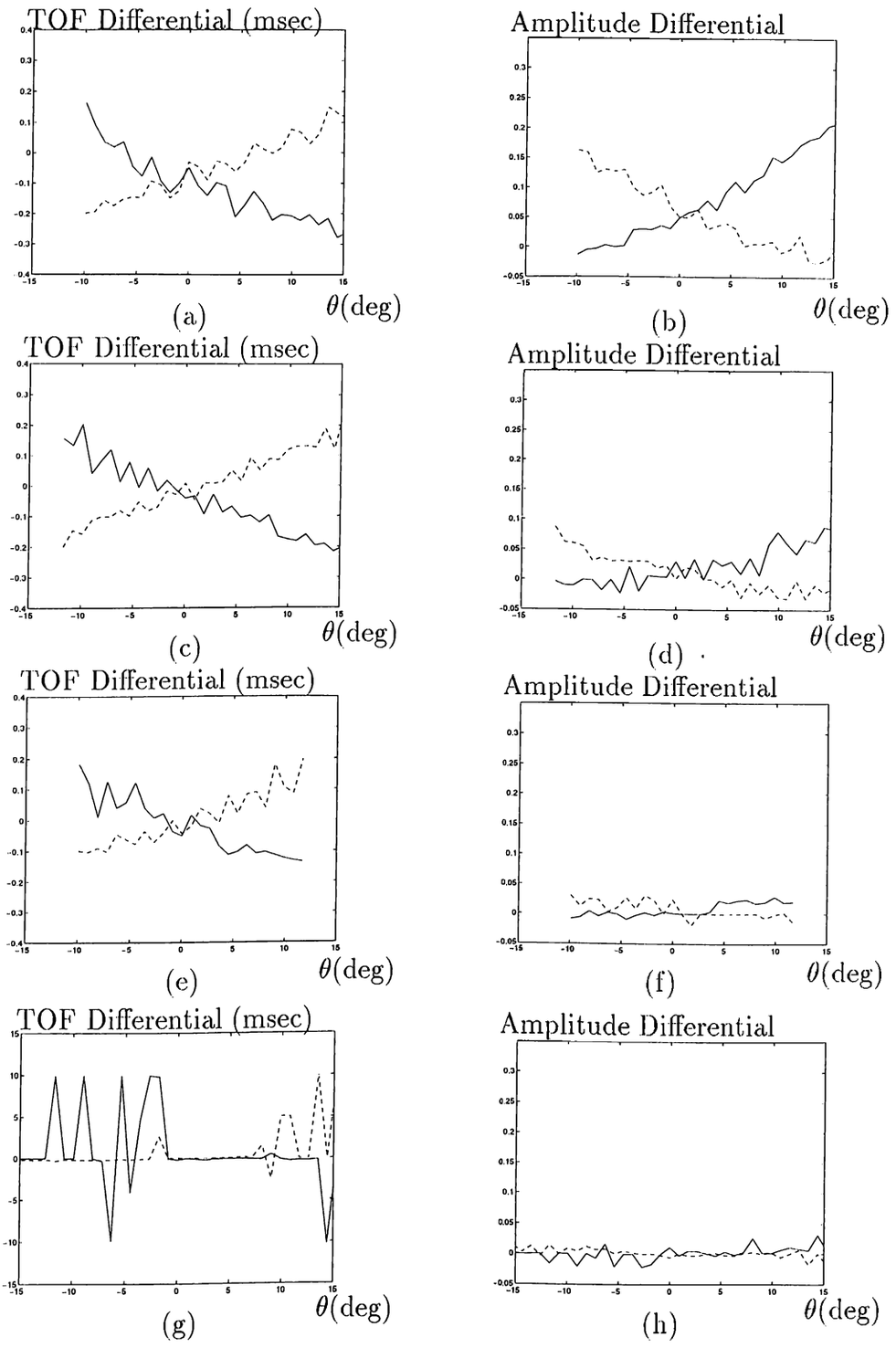


Figure 4.20: Differentials in TOF and amplitude characteristics of a cylinder with $r_c = 7.5$ cm when a Panasonic transducer pair with separation $d = 24$ cm is employed at the range values (a),(b) $r = 20$ cm (c),(d) $r = 40$ cm (e),(f) $r = 60$ cm (g),(h) $r = 80$ cm.

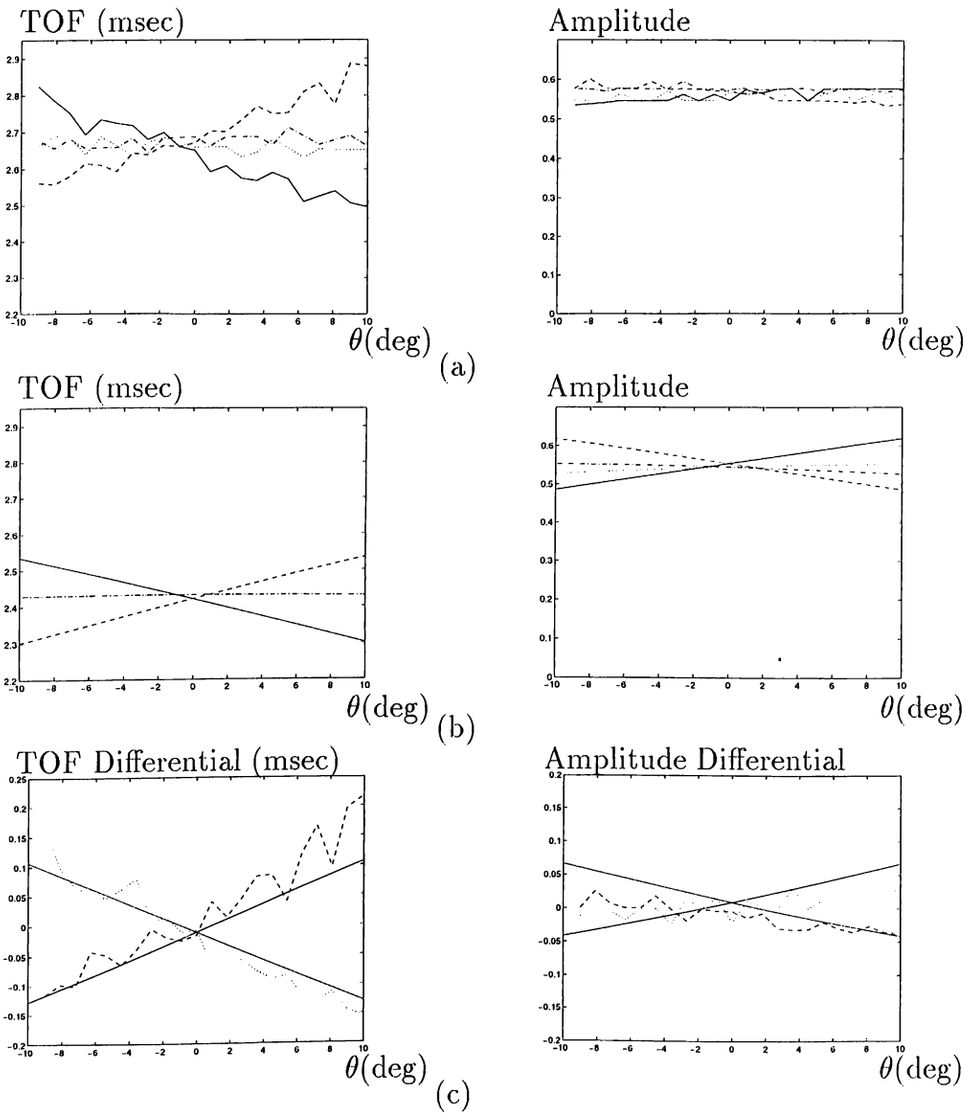


Figure 4.21: Comparison of TOF and amplitude characteristics of a cylinder of $r_c = 5.0$ cm with the theoretical predictions when a Panasonic transducer pair with separation $d = 24$ cm is employed at $r = 40$ cm (a) experimental results (b) theoretical results (c) differential signals.

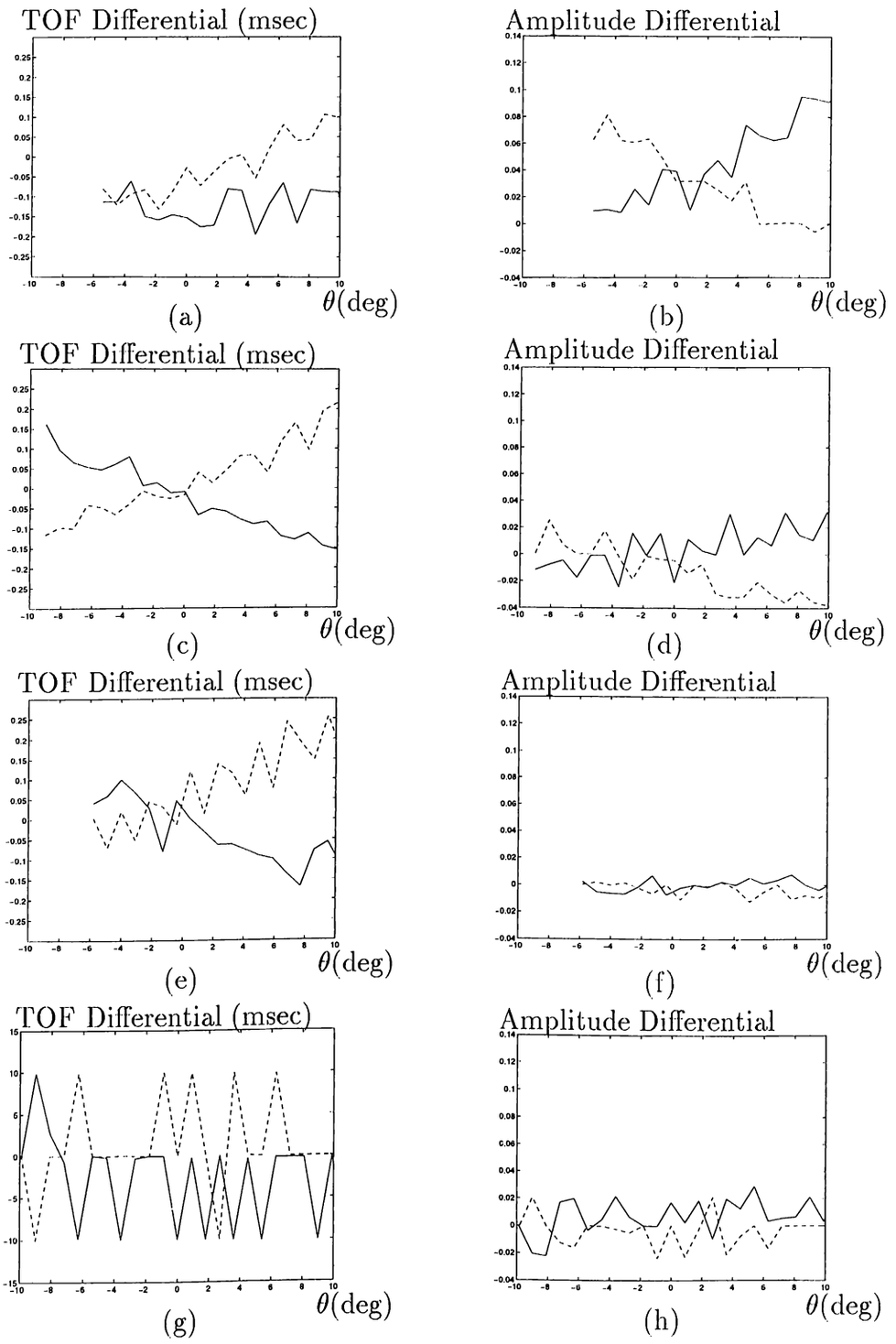


Figure 4.22: Differentials in TOF and amplitude characteristics of a cylinder with $r_c = 5.0$ cm when a Panasonic transducer pair with separation $d = 24$ cm is employed at the range values (a),(b) $r = 20$ cm (c),(d) $r = 40$ cm (e),(f) $r = 60$ cm (g),(h) $r = 80$ cm.

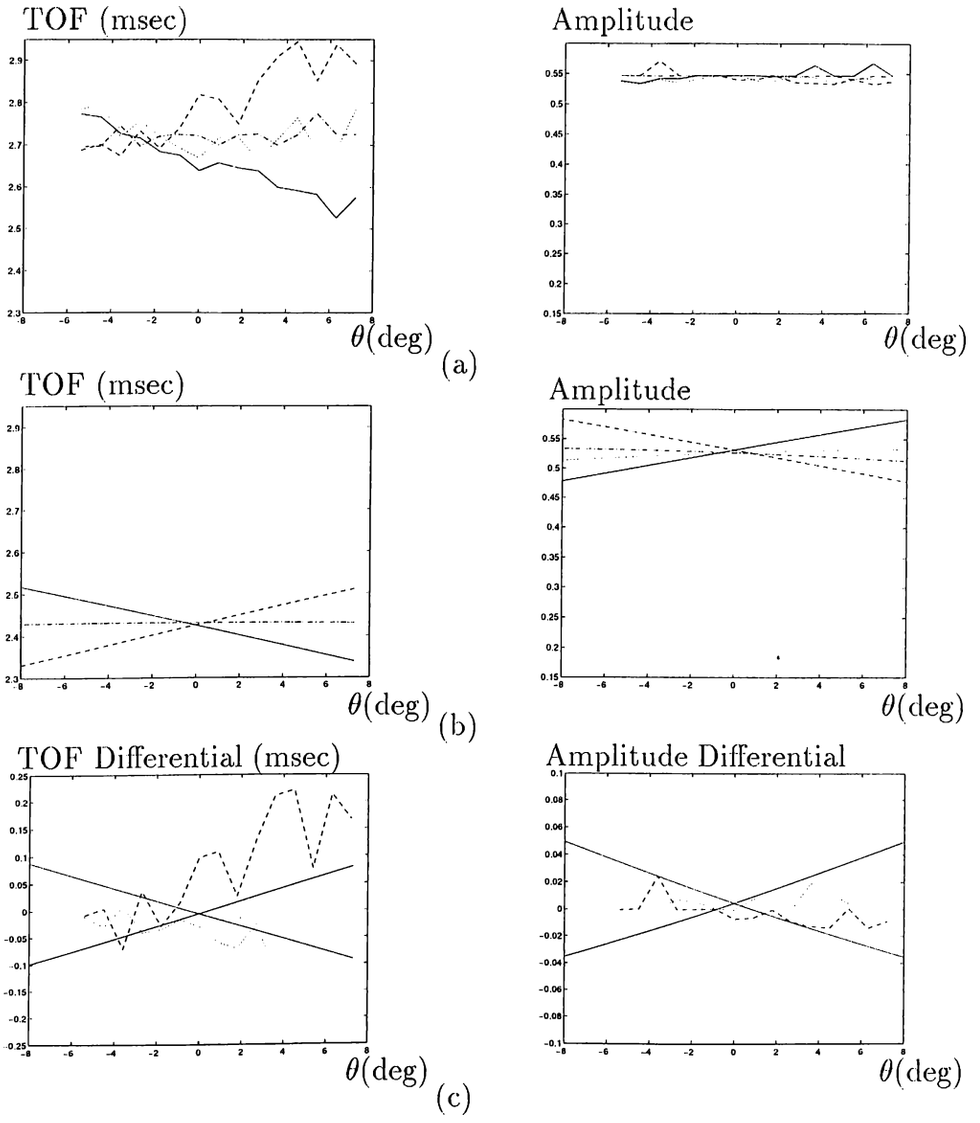


Figure 4.23: Comparison of TOF and amplitude characteristics of a cylinder of $r_c = 2.5$ cm with the theoretical predictions when a Panasonic transducer pair with separation $d = 24$ cm is employed at $r = 40$ cm (a) experimental results (b) theoretical results (c) differential signals.

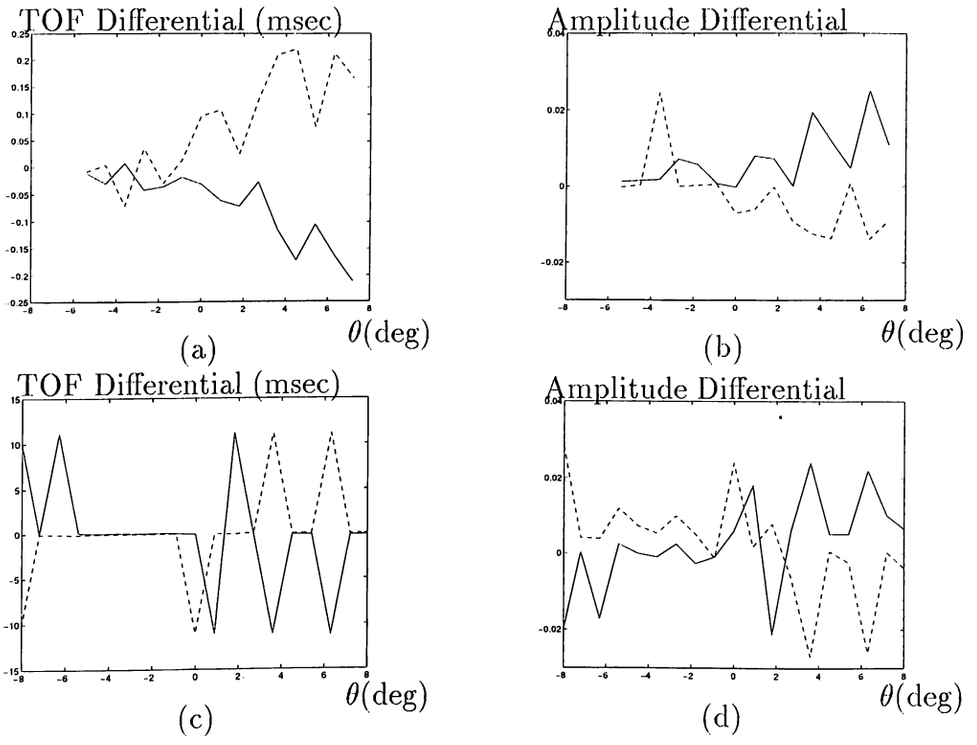


Figure 4.24: Differentials in TOF and amplitude characteristics of a cylinder with $r_c = 2.5$ cm when a Panasonic transducer pair with separation $d = 24$ cm is employed at the range values (a),(b) r =40 cm (c),(d) r =60 cm.

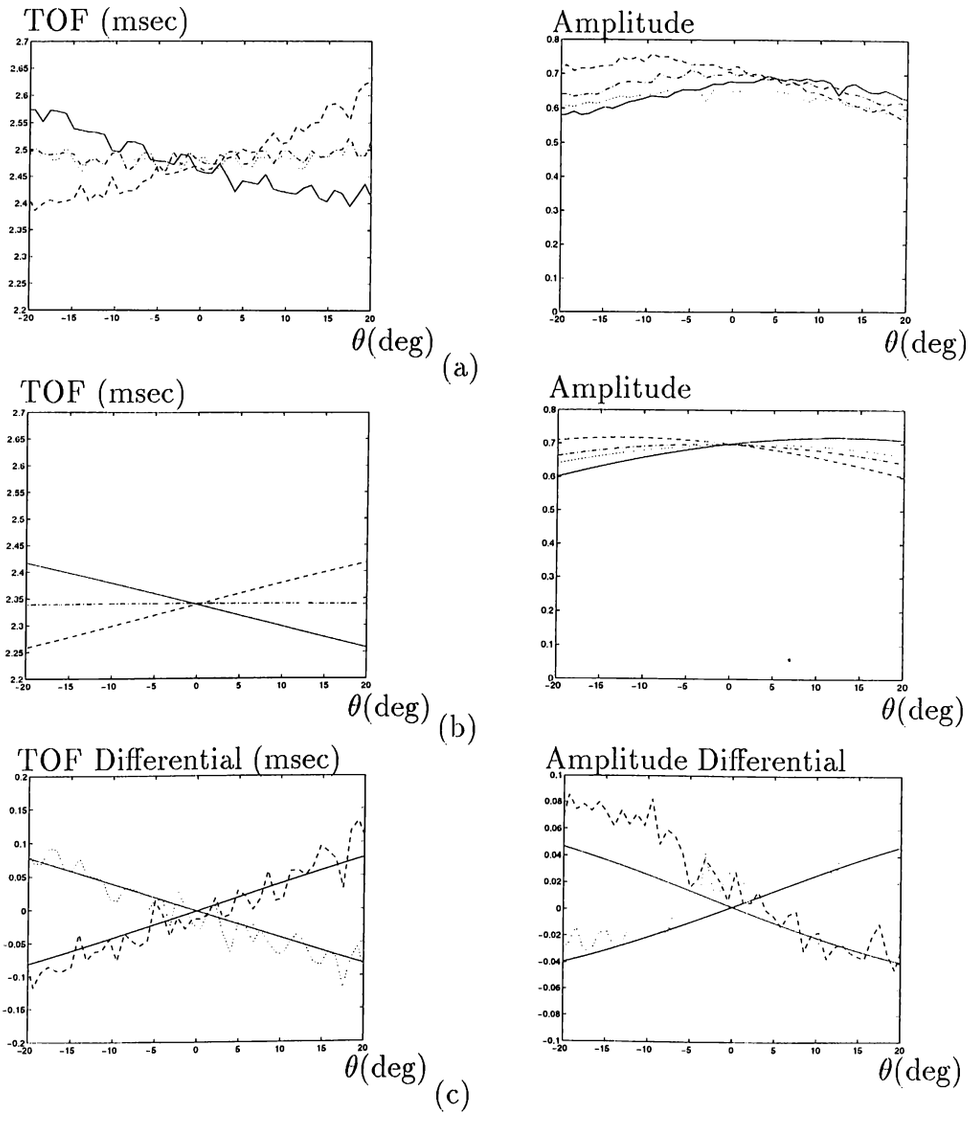


Figure 4.25: Comparison of TOF and amplitude characteristics of a cylinder of $r_c = 7.5$ cm with the theoretical predictions when a Panasonic transducer pair with separation $d = 8$ cm is employed at $r = 40$ cm (a) experimental results (b) theoretical results (c) differential signals.

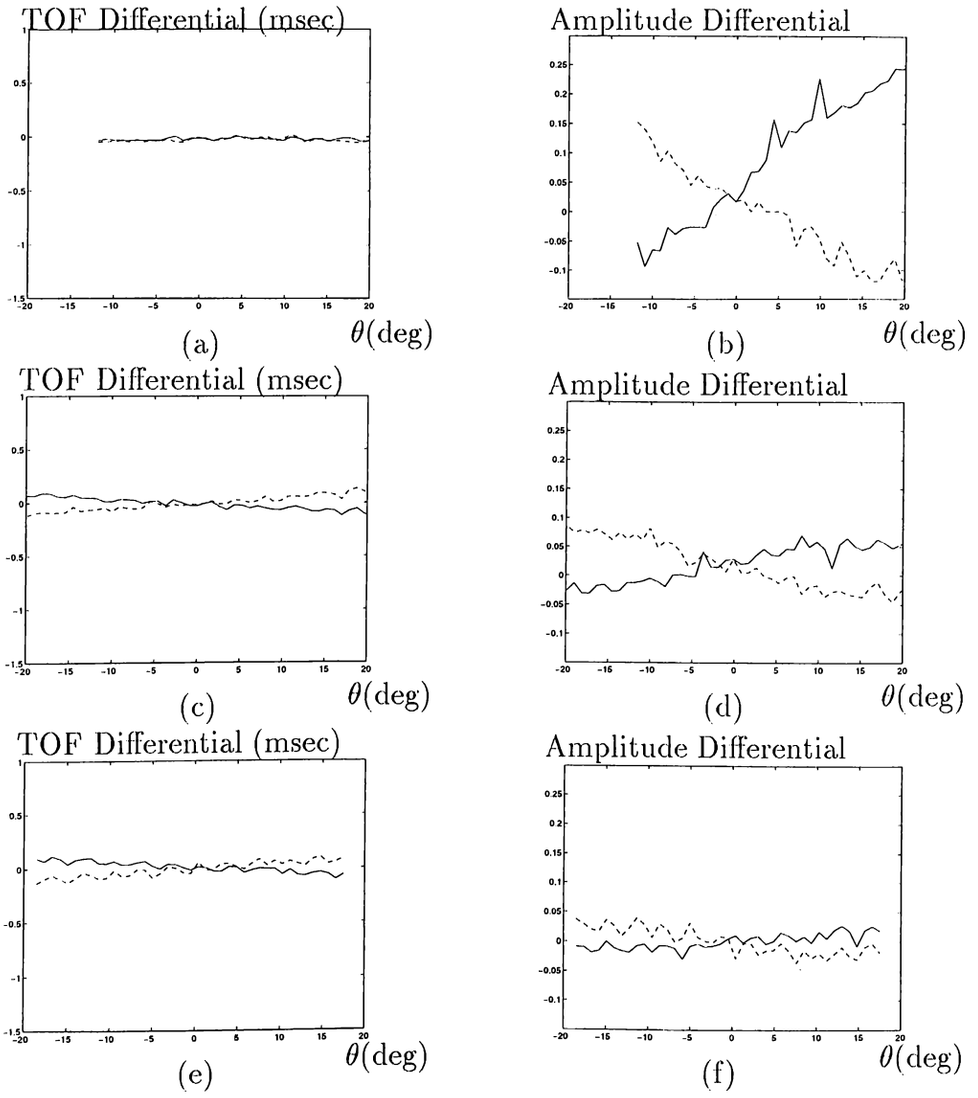


Figure 4.26: Differentials in TOF and amplitude characteristics of a cylinder with $r_c = 7.5$ cm when a Panasonic transducer pair with separation $d = 8$ cm is employed at the range values (a),(b) $r = 20$ cm (c),(d) $r = 40$ cm (e),(f) $r = 60$ cm.

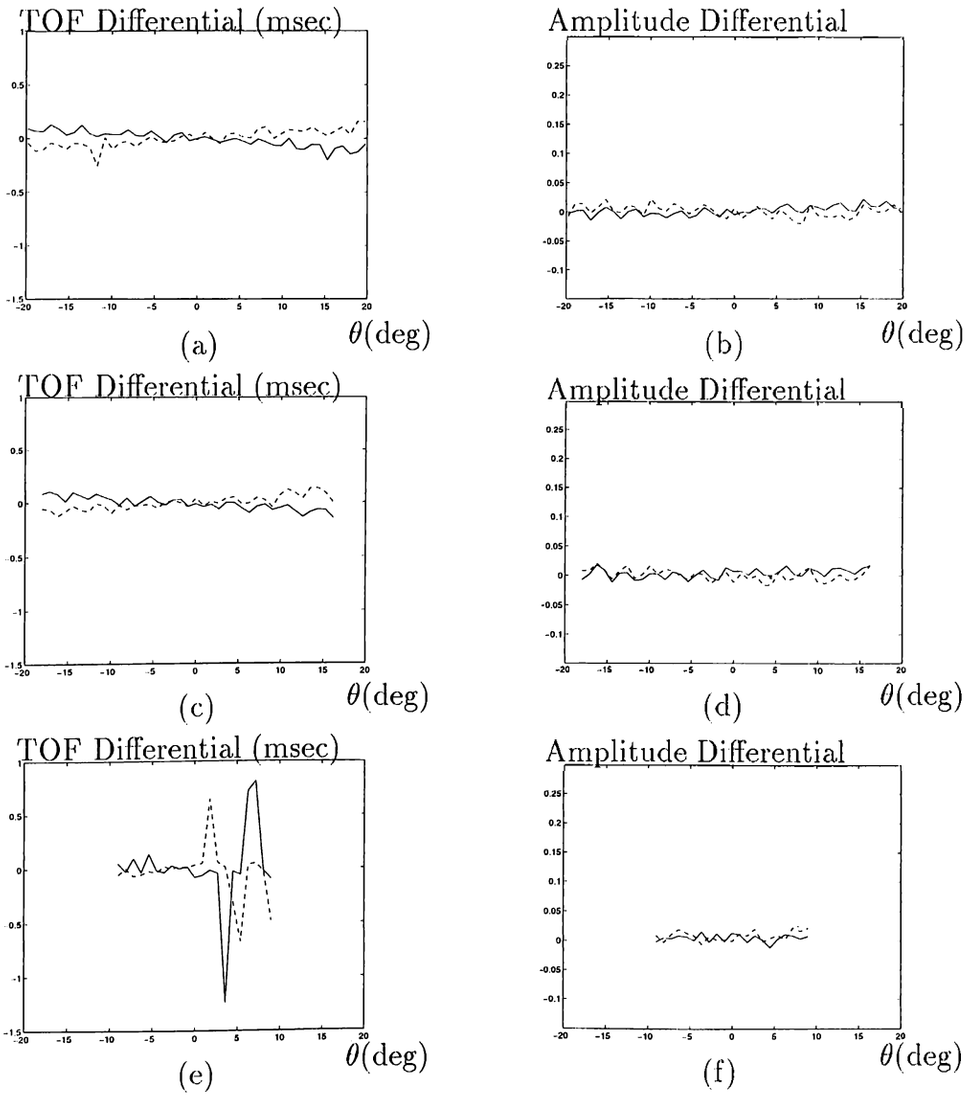


Figure 4.27: Differentials in TOF and amplitude characteristics of a cylinder with $r_c = 7.5$ cm when a Panasonic transducer pair with separation $d = 8$ cm is employed at the range values (a),(b) $r = 80$ cm (c),(d) $r = 100$ cm (e),(f) $r = 120$ cm.

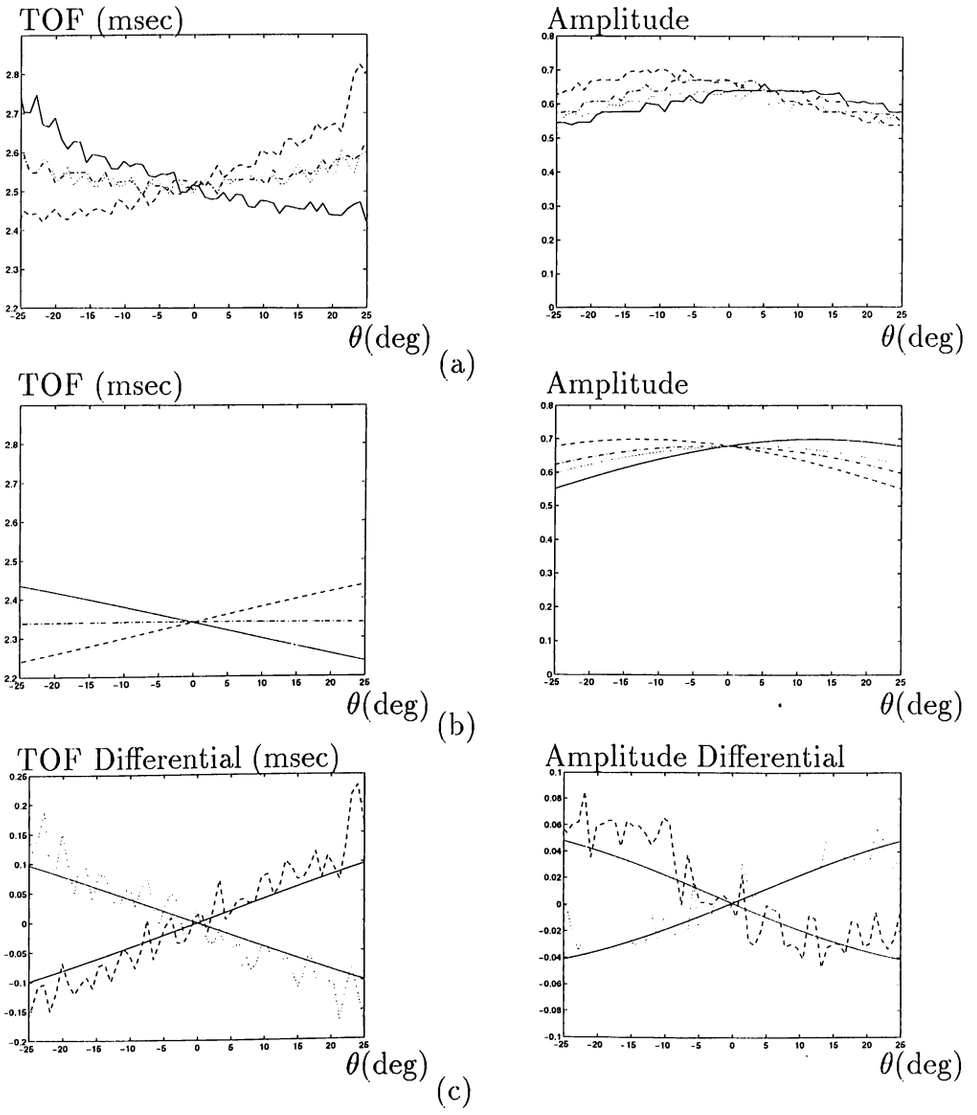


Figure 4.28: Comparison of TOF and amplitude characteristics of a cylinder of $r_c = 5.0$ cm with the theoretical predictions when a Panasonic transducer pair with separation $d = 8$ cm is employed at $r = 40$ cm (a) experimental results (b) theoretical results (c) differential signals.

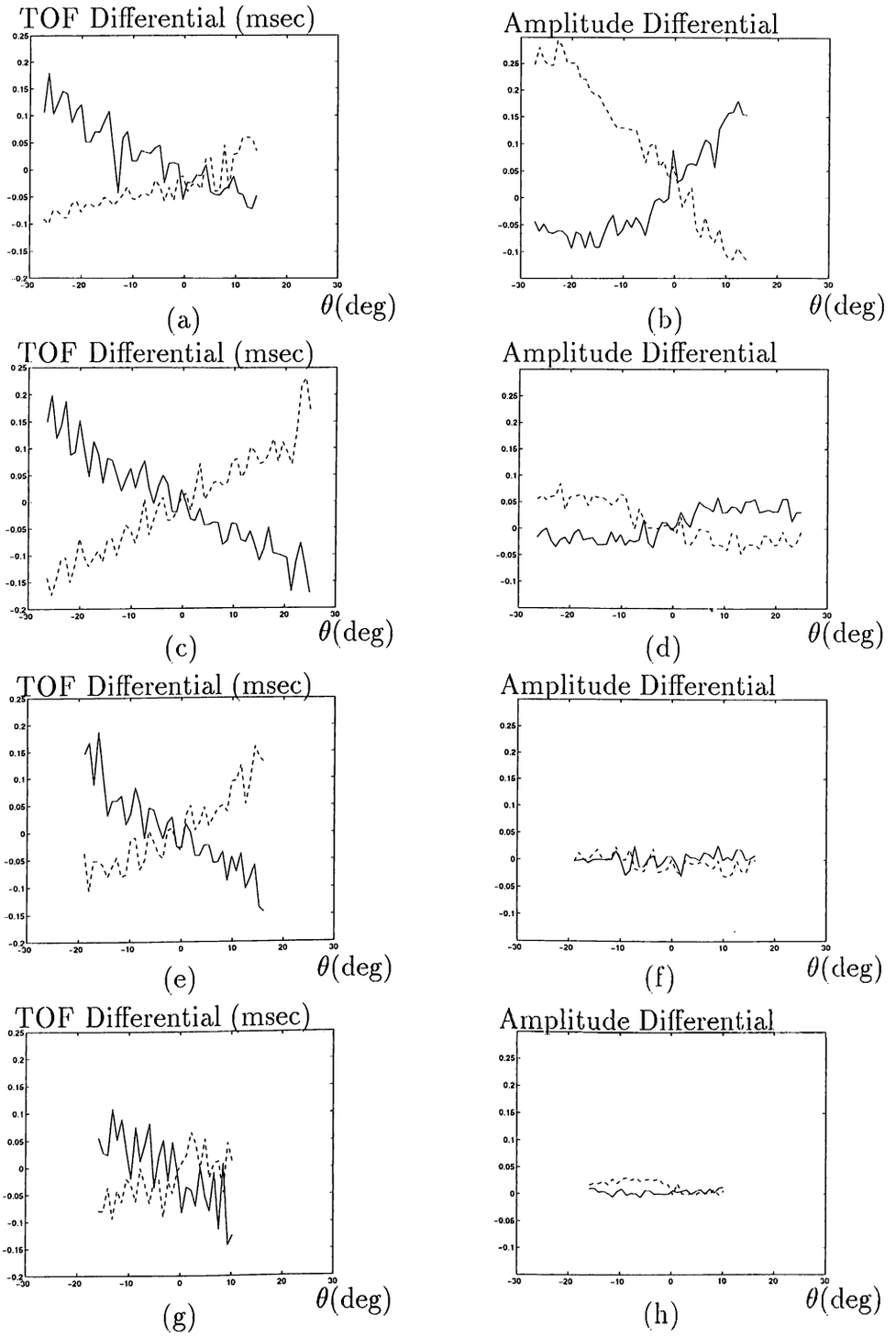


Figure 4.29: Differentials in TOF and amplitude characteristics of a cylinder with $r_c = 5.0$ cm when a Panasonic transducer pair with separation $d = 8$ cm is employed at the range values (a),(b) $r = 20$ cm (c),(d) $r = 40$ cm (e),(f) $r = 60$ cm (g),(h) $r = 80$ cm.

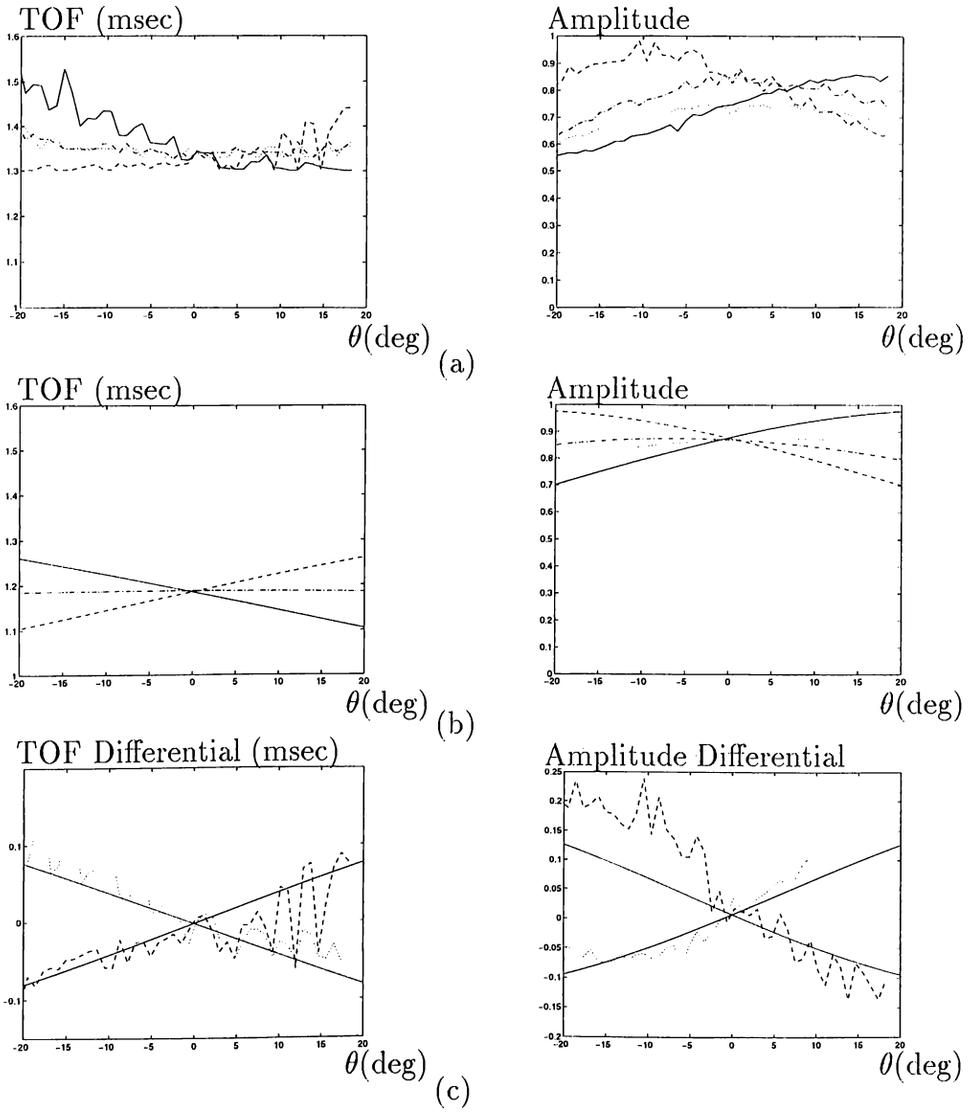


Figure 4.30: Comparison of TOF and amplitude characteristics of a cylinder of $r_c = 2.5$ cm with the theoretical predictions when a Panasonic transducer pair with separation $d = 8$ cm is employed at $r = 20$ cm (a) experimental results (b) theoretical results (c) differential signals.

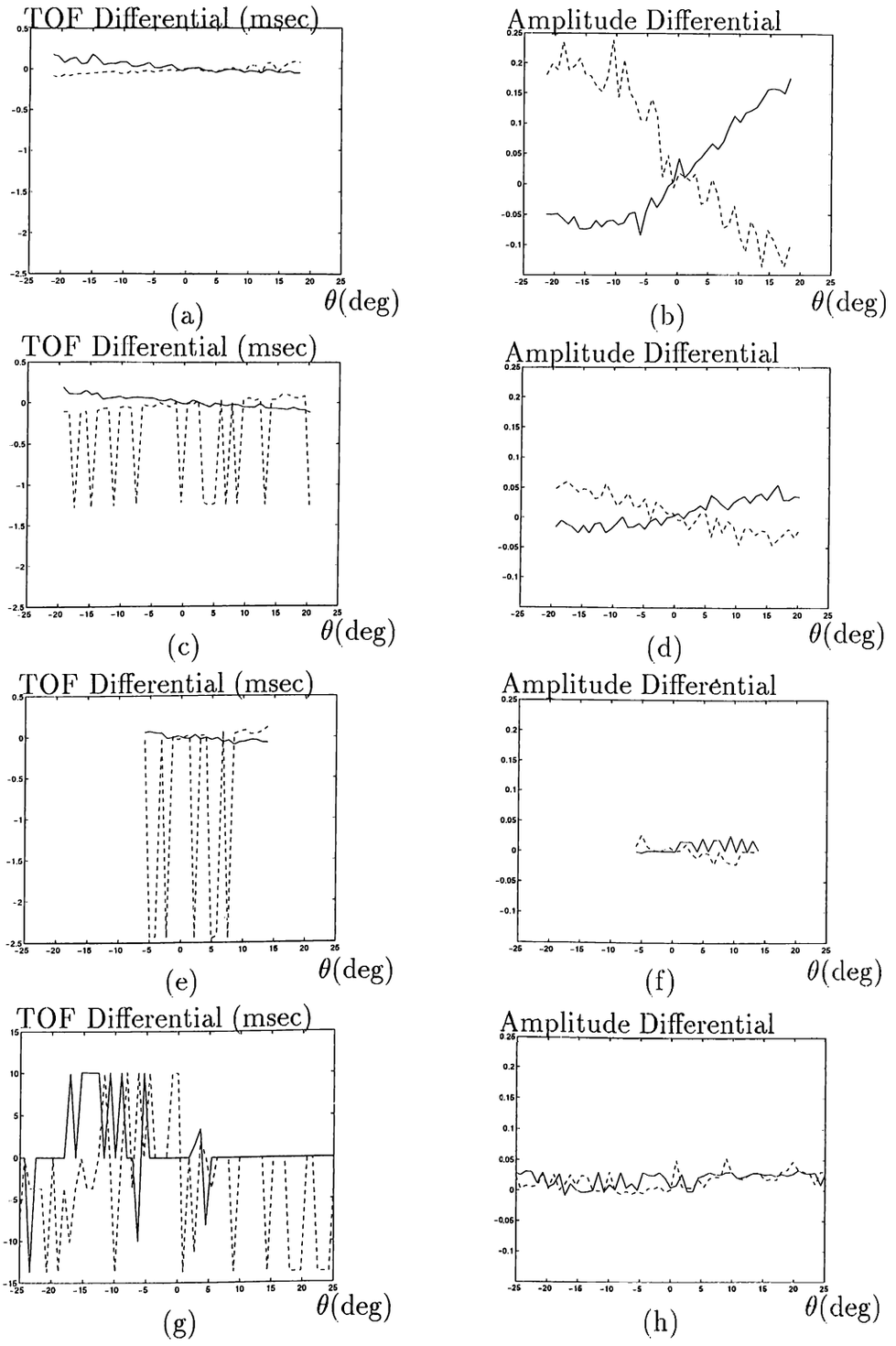


Figure 4.31: Differentials in TOF and amplitude characteristics of a cylinder with $r_c = 2.5$ cm when a Panasonic transducer pair with separation $d = 8$ cm is employed at the range values (a),(b) $r = 20$ cm (c),(d) $r = 40$ cm (e),(f) $r = 60$ cm (g),(h) $r = 80$ cm.

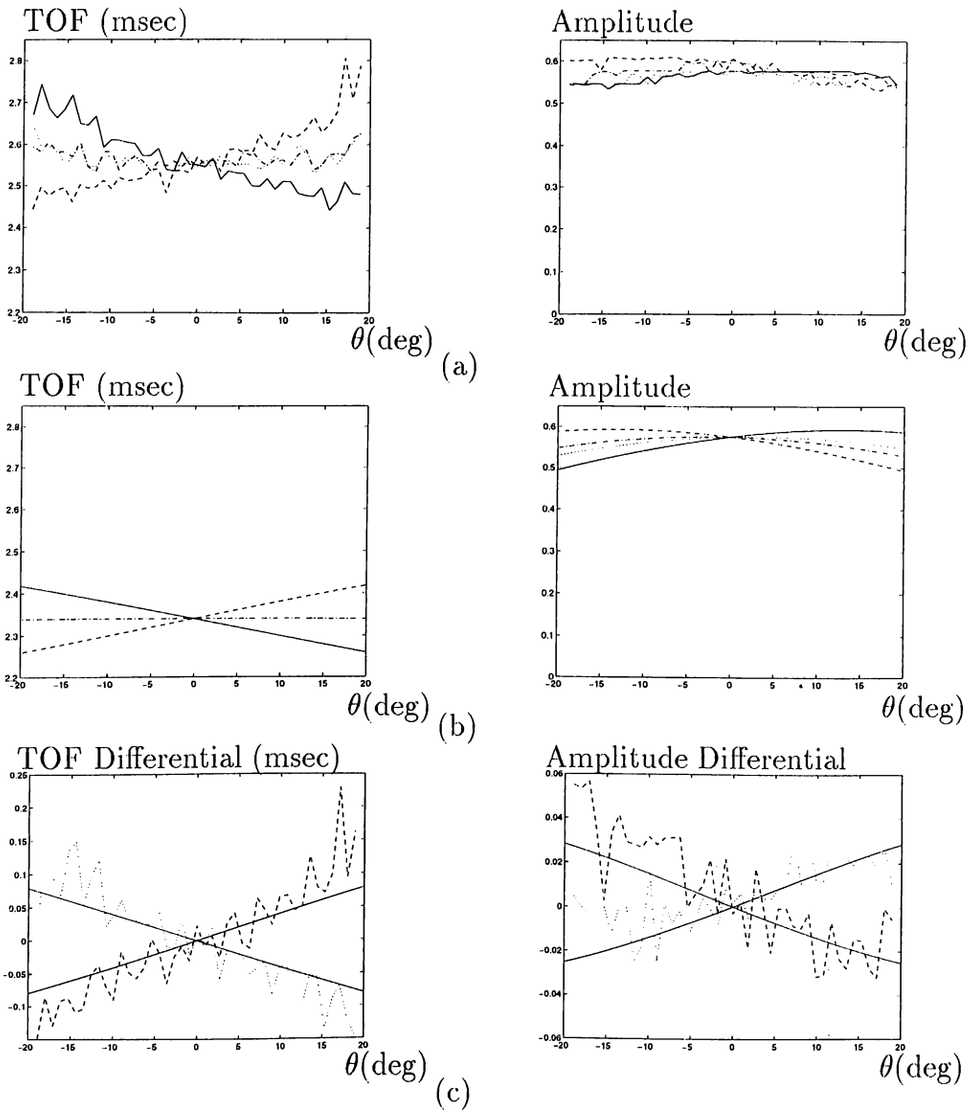


Figure 4.32: Comparison of TOF and amplitude characteristics of a cylinder of $r_c = 1.5$ cm with the theoretical predictions when a Panasonic transducer pair with separation $d = 8$ cm is employed at $r = 40$ cm (a) experimental results (b) theoretical results (c) differential signals.

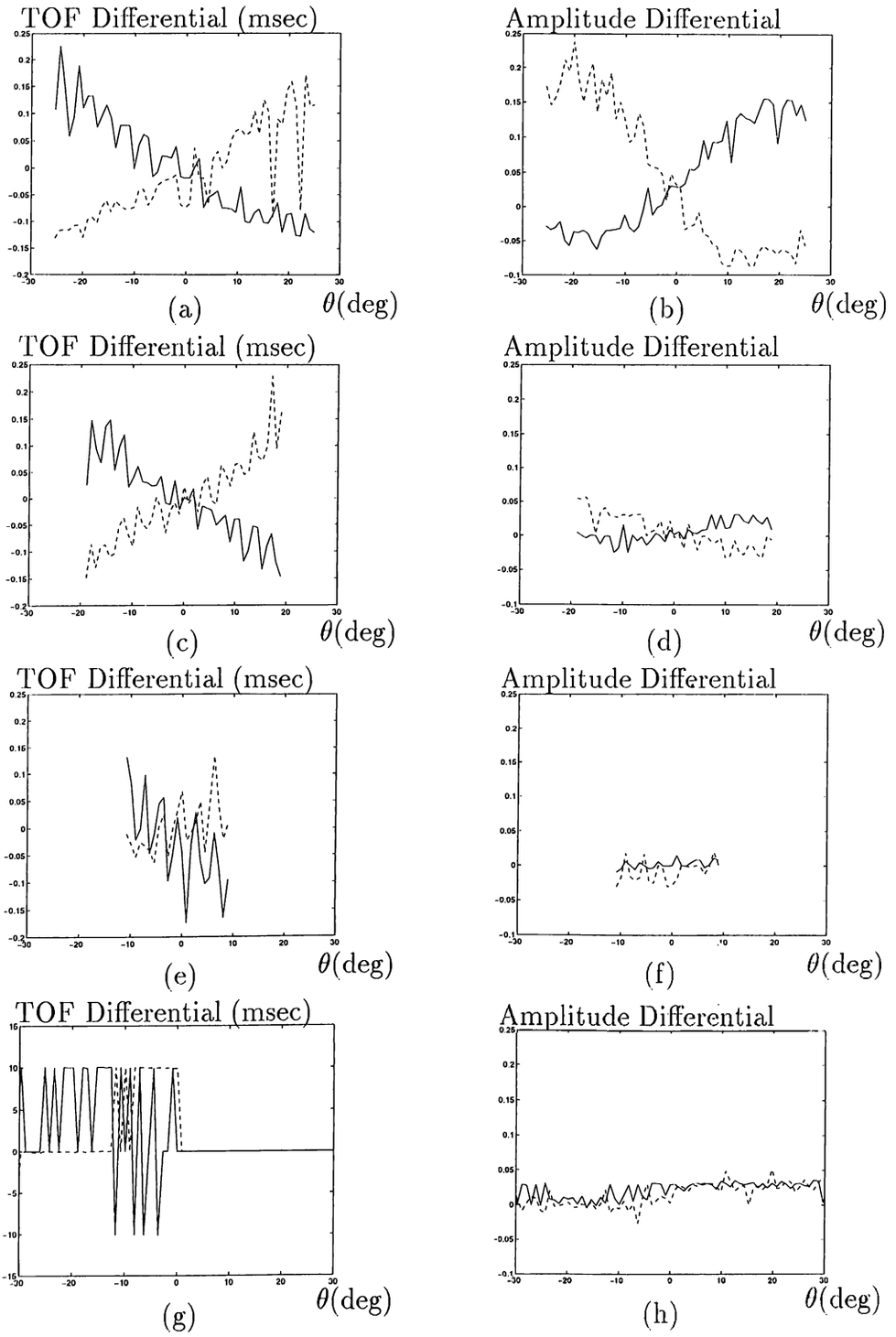


Figure 4.33: Differentials in TOF and amplitude characteristics of a cylinder with $r_c = 1.5$ cm when a Panasonic transducer pair with separation $d = 8$ cm is employed at the range values (a),(b) $r = 20$ cm (c),(d) $r = 40$ cm (e),(f) $r = 20$ cm (g),(h) $r = 40$ cm.

Belief assignment results to the TOF and amplitude characteristics of a plane at $r = 50$ cm when scanned with Panasonic transducers with separation $d = 24$ cm are given in Figure 4.34. In this figure, belief of being a planar type target primitive is greater than zero for $-20^o \leq \theta \leq 20^o$. Belief of being a plane and the belief of being an unknown target oscillate around 0.5 for $|\theta| \leq 10^o$, and the belief of being an unknown target is greater than the belief of a plane outside this region. Moreover, belief of being a corner or an acute corner is zero for all θ values. Estimated range and azimuth values are given in Figure 4.35. Referring to this figure, maximum range error is 0.5 cm and maximum error in the azimuth estimate is 0.7^o.

Beliefs to the TOF and amplitude characteristics of a corner at $r = 80$ cm when scanned with the Panasonic sensing system with separation $d = 24$ cm are assigned as shown in Figure 4.36. In this figure, although the belief of being an acute corner is around one for $|\theta| \leq 5^o$, estimated angle of this acute corner is around 90^o in this region (Figure 4.37(c)). Therefore, the final decision is a corner. The belief of being a corner is greater than the belief of being an unknown target for all θ values outside this region except one point at which the belief of being a plane is one. However, belief of being a plane is zero at all θ values except this point. Estimated range and azimuth values are given in Figure 4.37. Referring to this figure, maximum range error is 0.3 cm and the maximum error in azimuth estimates is 3.6^o in the region $\theta \in [-5^o, 5^o]$.

Beliefs assigned to the TOF and amplitude characteristics of an acute corner of $\theta_c = 60^o$ at $r = 40$ cm which is scanned with the same system are given in Figure 4.38. In this figure, belief of being an acute corner is always greater than the belief of being an unknown target and belief of being a plane or a corner is always zero. Estimated range, azimuth and wedge angle of acute corner are given in Figure 4.39. Referring to this figure, maximum range error is 2.0 cm, maximum error in azimuth estimates is 3.0^o and maximum error in estimated angle of the acute corner is 4.2^o for the interval $\theta \in [-6^o, 10^o]$.

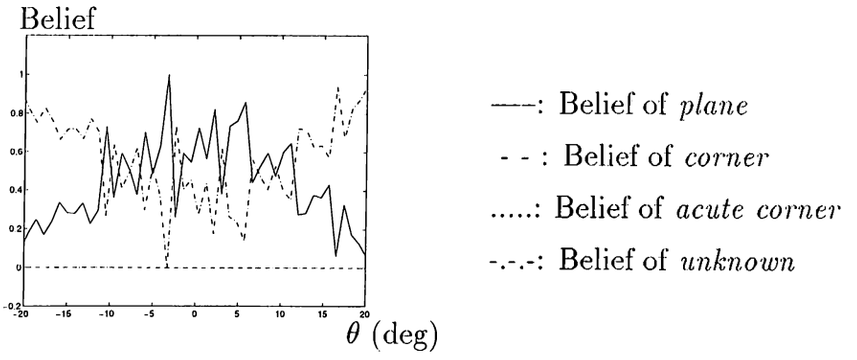


Figure 4.34: Belief assignment to a plane at $r = 50$ cm scanned with a Panasonic transducer pair with separation $d = 24$ cm.

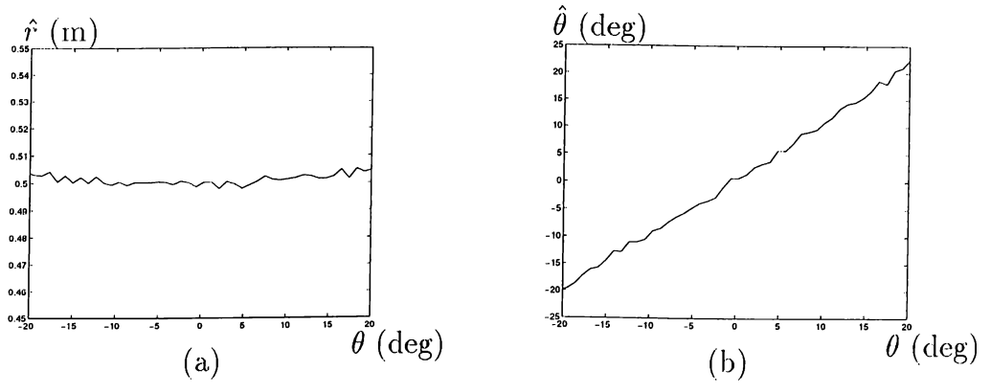


Figure 4.35: Estimated (a) range and (b) azimuth values of a plane at $r = 50$ cm scanned with a Panasonic transducer pair with separation $d = 24$ cm.

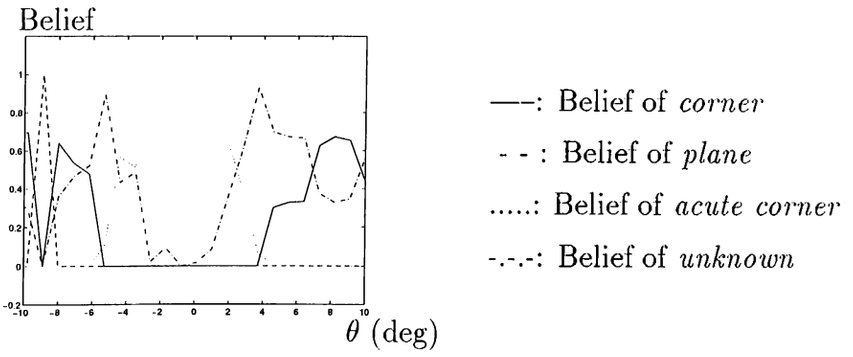


Figure 4.36: Belief assignment to a corner at $r = 80$ cm which is scanned with Panasonic transducers with separation $d = 24$ cm.

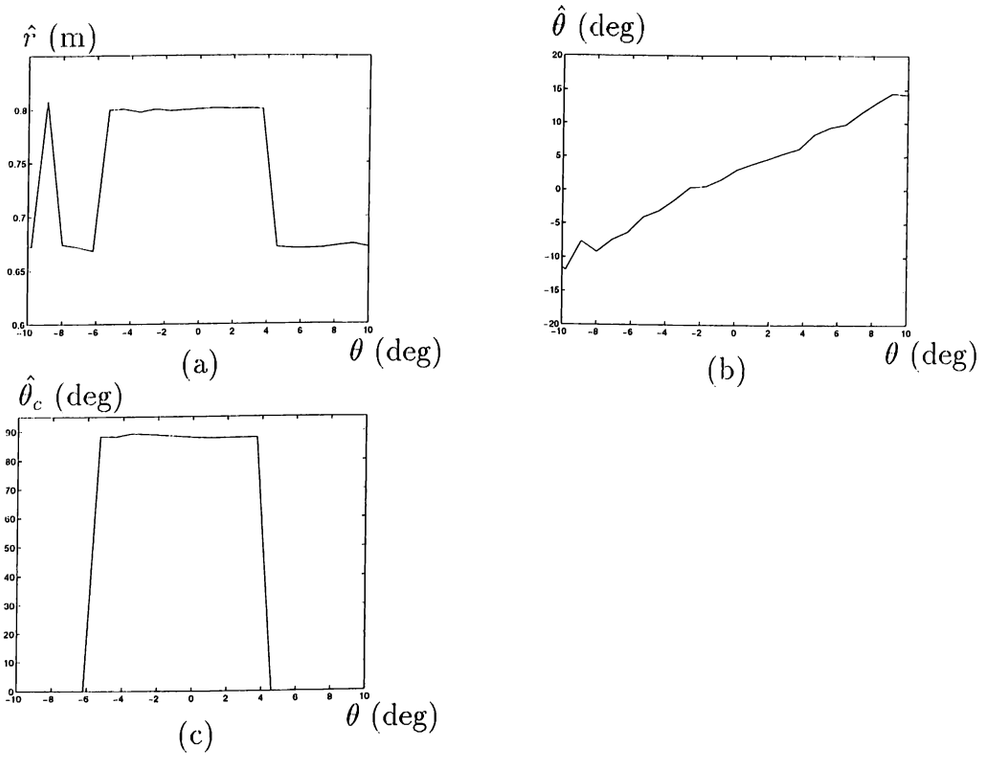


Figure 4.37: Estimated (a) range (b) azimuth and (c) wedge angle of a corner at $r = 80$ cm scanned with Panasonic transducers with separation $d = 24$ cm.

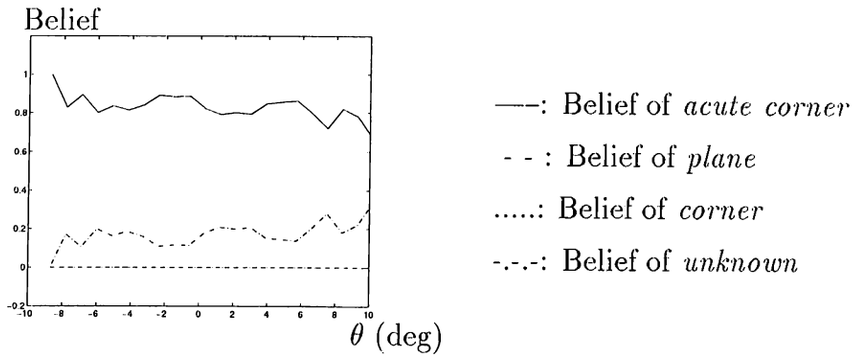


Figure 4.38: Belief assignment to an acute corner of $\theta_c = 60^o$ at $r = 40$ cm scanned with a Panasonic transducer pair with separation $d = 24$ cm.

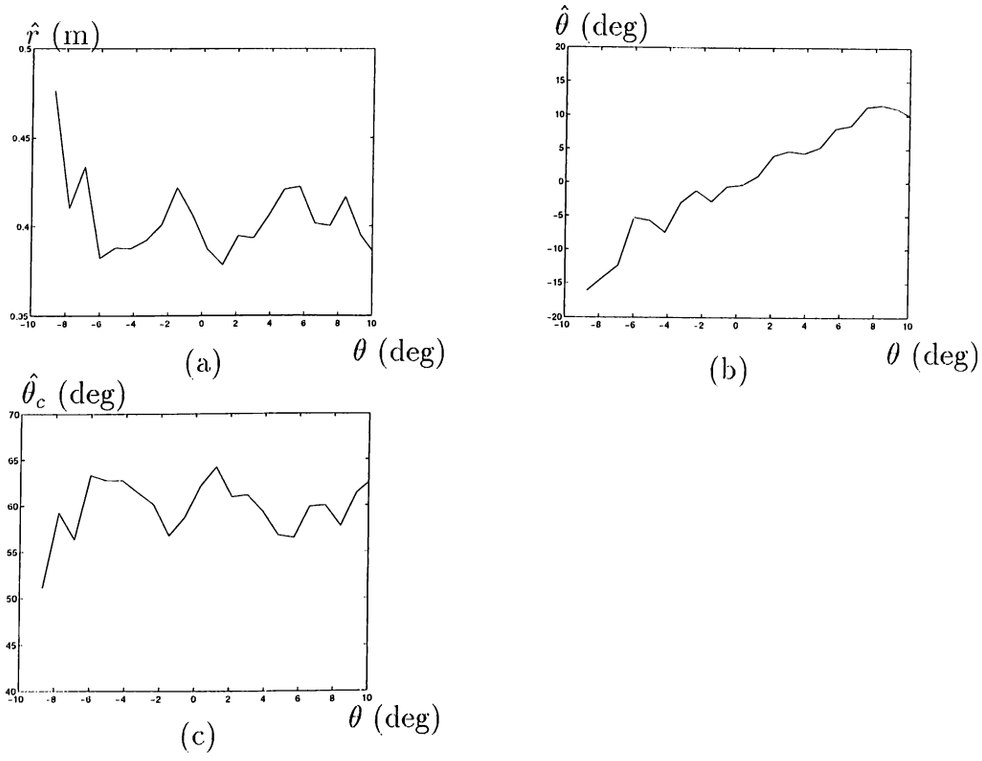


Figure 4.39: Estimated (a) range (b) azimuth and (c) angle of an acute corner of $\theta_c = 60^o$ at $r = 40$ cm scanned with a Panasonic transducer pair with separation $d = 24$ cm.

The results are tested further in an uncluttered rectangular room with specular surfaces. A 1.38m×1.15m rectangular room is scanned with three sensors pairs located at (0.0,0.0), $(-0.21, 0.17)$ and (0.35,0.29) in meters. Due to the physical limitations with the hardware, the sensors cannot cover the whole range of ϕ but rotate over the range $0^o \leq \phi \leq 284^o$. The range readings of the transducer pair located at $(-0.21, 0.17)$ are given in Figure 4.40 as an example.

Beliefs are assigned to the TOF and amplitude characteristics reflected from corners and planar walls by this transducer pair at each location. These belief values are fused by Dempster-Shafer rule of combination. The results of belief assignment for each transducer pair are given in Figure 4.41. Referring to this figure, with the sensor located at (0.0,0.0), the corners 1 and 3 can be detected correctly but the corner 2 and the walls of the room cannot be differentiated accurately. With the pair at $(-0.21, 0.17)$, corners 1, 2 and 3 and planes 1 and 2 are detected correctly, but the planes 3 and 4 are detected as corner since differentials in the amplitude of echo which is reflected by planes cannot be detected at this transducer pair accurately when $r \geq 50$ cm. With the pair at (0.35,0.29), planes and corners closest to the transducer pair are detected very well but the two furthest planes are detected as corner due to the same reason as explained above.

Pairwise fusion of the beliefs assigned by the transducer pair at each location are given in Figure 4.42(a)-(c). Referring to this figure, when the beliefs assigned by transducer pairs at locations (0.0,0.0) and $(-0.21, 0.17)$ are fused, recognition of corners 1 and 3 and plane 2 is improved. When fusion is done for the beliefs assigned by the transducer pairs located at (0.0,0.0) and (0.35,0.29), classification of corners 1 and 3 and planes 2 and 3 is improved. However, belief of being a corner at the position of plane 1 increases since both transducer pairs detect plane 1 as corner. Identification of corners 1, 2 and 3 and planes 2 and 3 are improved when the beliefs assigned by the transducer pairs located at $(-0.21, 0.17)$ and (0.35,0.29) are fused. Therefore, the best result for pairwise fusion is obtained when the beliefs assigned by the transducer pairs at these locations are fused.

When the beliefs assigned by all three sensors are fused by Dempster-Shafer rule of combination, corners 1 and 3 and plane 2 are detected very well. Corner 2 and plane 3 are detected accurately except for a few values of ϕ. However, plane 1 and plane 4 are detected as corner since the sensors at the locations (0.0,0.0) and (0.35,0.29) detect these planes as corner. Therefore, belief of being a corner is strengthened (4.42(d)).

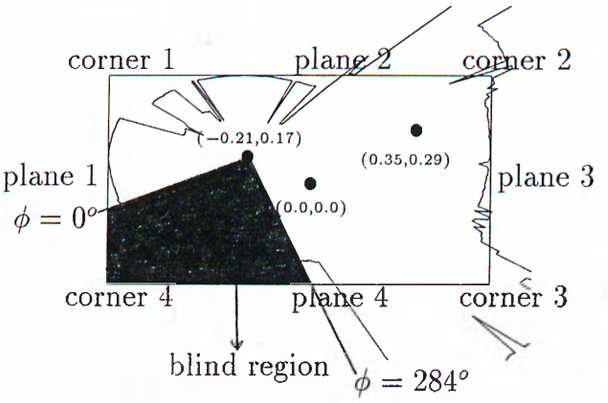


Figure 4.40: Range readings of the sensor located at (- 0.21, 0.17) in a rectangular room.

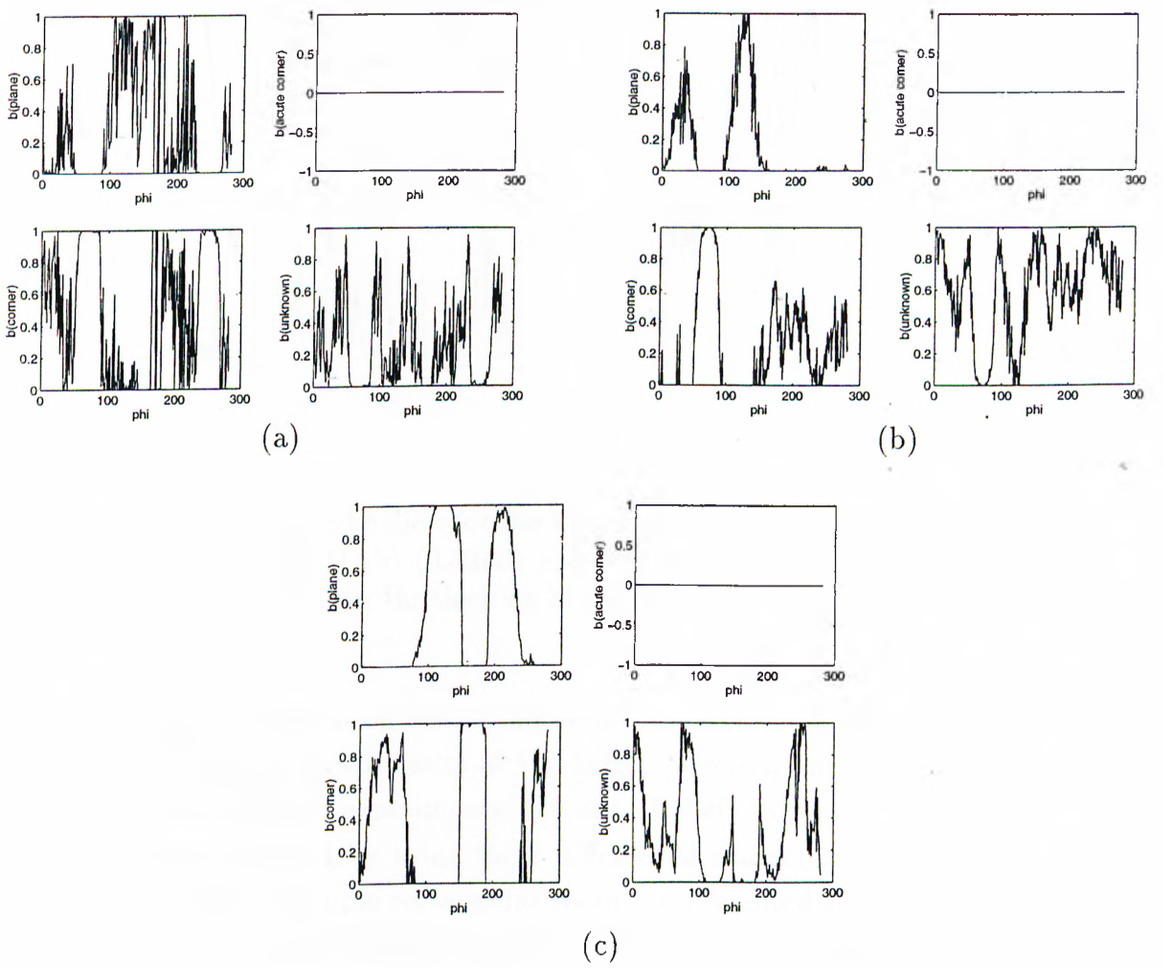


Figure 4.41: Belief assignment for the sensors located at (a) (0.0,0.0) (b) (−0.21, 0.17) (c) (0.35,0.29).

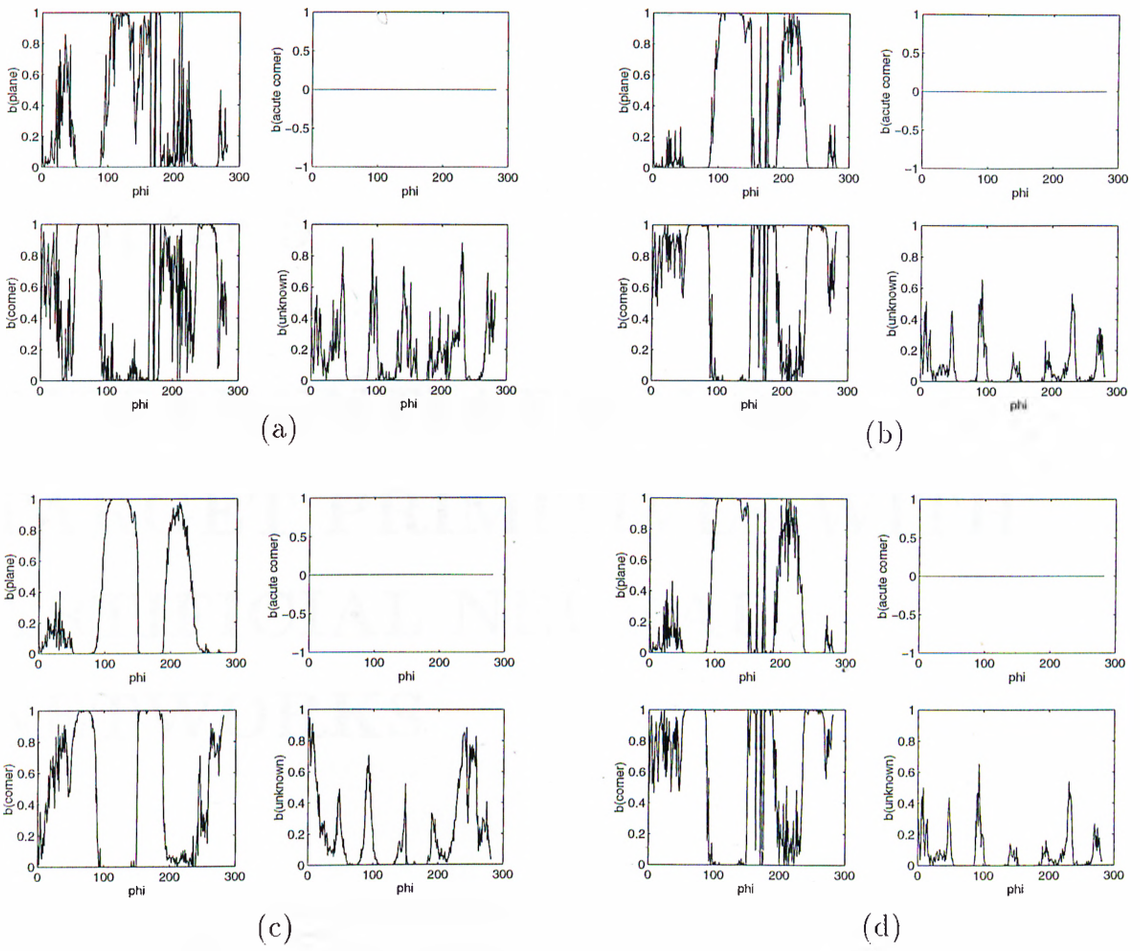


Figure 4.42: Pairwise fused beliefs for the Panasonic transducers located at (a) (0.0,0.0) and $(-0.21, 0.17)$ (b) (0.0,0.0) and (0.35,0.29) (c) $(-0.21, 0.17)$ and (0.35,0.29) (d) Results when the decision of all three pairs are fused.

In this chapter, simulation results are verified by obtaining differentials in TOF and amplitude characteristics of the target primitives at various range values with two different experimental set-ups. Beliefs to plane, corner and acute corner are assigned by using these differential characteristics. Experimental results from a simple rectangular room are presented to demonstrate the fusion process using $1-3$ sensor pairs. In the next chapter, data fusion from multiple sonars will be done with artificial neural network in the differentiation of target primitives.

Chapter 5

DIFFERENTIATION OF TARGET PRIMITIVES WITH ARTIFICIAL NEURAL NETWORKS

In this chapter, a brief introduction about *neural networks* is given. As a second approach, classification of target primitives with neural networks is introduced.

5.1 Introduction to Artificial Neural Networks

Artificial neural networks are computing structures which consist of massively parallel interconnections of simple computing elements which are called *neurons* [36]. Their importance comes from the fact that the brain is much better than computers in pattern recognition since humans can recognize objects of all shape and orientations under a wide range of conditions. Objects in unstructured environments under poor lighting conditions even when they are occluded by other objects are recognized by human beings without much effort.

The neurons in a layered neural network are distributed in several distinct layers. The interfacing layer on the input side is called *the input layer* and the layer on the output side is called *the output layer*. All the intermediate layers

are called *hidden layers.* All neurons in a layer are connected to each neuron in the adjacent layers with a connection strength which is represented in the form of a weight value which acts as a signal multiplier on the corresponding connection link. Most training algorithms can adapt these weights in time to improve performance of the network based on current results. The input to a neuron in a layer is the weighted linear summation of all the incoming signals on the various connection links. Moreover, at the input of a neuron, this summation is compared to a threshold value which is called the *bias* and the difference drives an output function called an *activation function.* Commonly used activation functions are sigmoid and hyperbolic tangent functions shown in Figure 5.1.

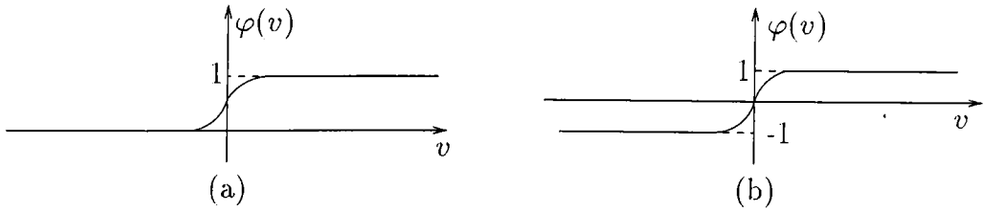


Figure 5.1: The two most common choices for the activation function: (a) sigmoid (b) hyperbolic tangent functions.

The artificial neural networks are used efficiently as pattern classifiers in many applications. The function of these pattern classifiers is the mapping of input data into one of the decision classes. Neural networks for pattern classification is designed by constructing the network by applying training procedures which computes the boundaries of each decision region. The most frequently used training procedure for these networks is the backpropagation algorithm which is a gradient-descent procedure to minimize the error at the output [37]. The solutions which represent local minima are occasionally obtained with the backpropagation algorithm due to the gradient search procedure used in this algorithm. Some other training procedures are Boltzmann learning [38], counter propagation [39] and Madaline Rule II [40]. These training procedures, including backpropagation algorithm, are slow. An important parameter which significantly affects the learning rate and overall classification performance is the number of neurons needed in the hidden layers. This parameter cannot be specified by these training procedures. If the network is not sufficiently large, the training procedures will never converge. Alternatively, if the number of hidden layers and neurons in these layers are unnecessarily high, this introduces unnecessary computational load. Although a single neuron is capable of only a linear mapping, a network with multiple hidden layers provides any desired mapping.

These classifiers are non-parametric and make weaker assumptions on the shape of underlying distributions of input data than any traditional statistical classifiers. Therefore, they are more robust when distributions are non-Gaussian and generated by nonlinear processes.

5.2 Target Classification with Artificial Neural Networks

In this section, six neural networks are designed to classify the target primitives which are: plane, corner, edge of $\theta_e = 90^o$, cylinder with $r_c = 15$ cm, acute corners of $\theta_c = 30^o$, $\theta_c = 45^o$ and $\theta_c = 60^o$. First five networks are designed by training with the amplitude and TOF patterns of these target primitives at a *specific* range value r. However, the last one is designed by training the amplitude and TOF patterns at *five* different range values at the same time. These range values are: 0.5, 1.0, 1.5, 2.0 and 2.5 m. These networks were trained by using the error back propagation algorithm [37].

In this case, data which is taken from a single sensor pair has the same structure as in the previous chapters. Each target primitive at range value r is scanned for $-13.5^o \leq \theta \leq 13.5^o$ with motor stepping angle 0.9^o. One can also think of this system as 31 sensor pairs, each located at range value r and angle θ_i where

$$\theta_i = -13.5 + 0.9(i-1) \quad i = 1, 2, ..., 31 \tag{5.1}$$

In this case, unlike the belief function approach, *measurements* of these sensor pairs are fused with the neural network, not their decisions. The neural network which is used in this study has an input layer, two hidden layers and an output layer (Figure 5.2). The reason for choosing a multilayer network is the nonlinearity of the input patterns which yields nonlinear classification regions. Using the same training set, neural network models with a single hidden layer of neurons whose number range between 20 and 100 were taught. However, since edge and cylinder target primitives can only be distinguished at a single point which is $\theta = 0^o$, these networks become trapped in local minima. As a result, number of hidden layers are increased to two. Again neural networks with different number of neurons in these layers were trained. The network with the smallest size which learned these target primitives at *specific* range value r is the network with 50 neurons in the first hidden layer and 25 neurons in the

second hidden layer. Although many networks with various number of hidden layers and different number of neurons in these layers were trained using the training set which consists of the amplitude and TOF patterns of these target primitives at five different range values at the same time, all networks became trapped in local minima since most of these networks can not differentiate edges and cylinders. Therefore a network of two hidden layers with 50 and 25 neurons in first and second layers respectively, is used. This network is the only network in these trained networks which cannot learn only one pattern which is a cylinder of $r_c = 15$ cm at range $r = 1.5$ m. For this network, an average error of 0.002403 is obtained at the 164117^{th} cycle. Here, the average error is the average of the Euclidean distance between output of the network at a cycle and desired one for all training patterns:

$$E_{ave} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{2} ||d_i - o_i||^2 \tag{5.2}$$

where N is the number of training patterns, d_i is the desired output for the i^{th} pattern and o_i is the output of the network for this pattern.

For finding the weight values of these neural networks, PlaNet is used as a software tool and related computer program is provided in Appendix F (Program 11). The number of cycles until the networks learns the training patterns and the average error which is obtained over these cycles are given in Table 5.1. The number of training patterns for each network is also provided in this table.

r(m)	N	number of cycles	E_{ave}
0.5	7	770	10^{-4}
1.0	7	1226	10^{-4}
1.5	7	6430	10^{-4}
2.0	7	79600	10^{-4}
2.5	7	92488	10^{-4}
0.5–2.5	35	164117	0.002403

Table 5.1: Number of learning cycles and the average error for the network which is trained with the patterns at range r.

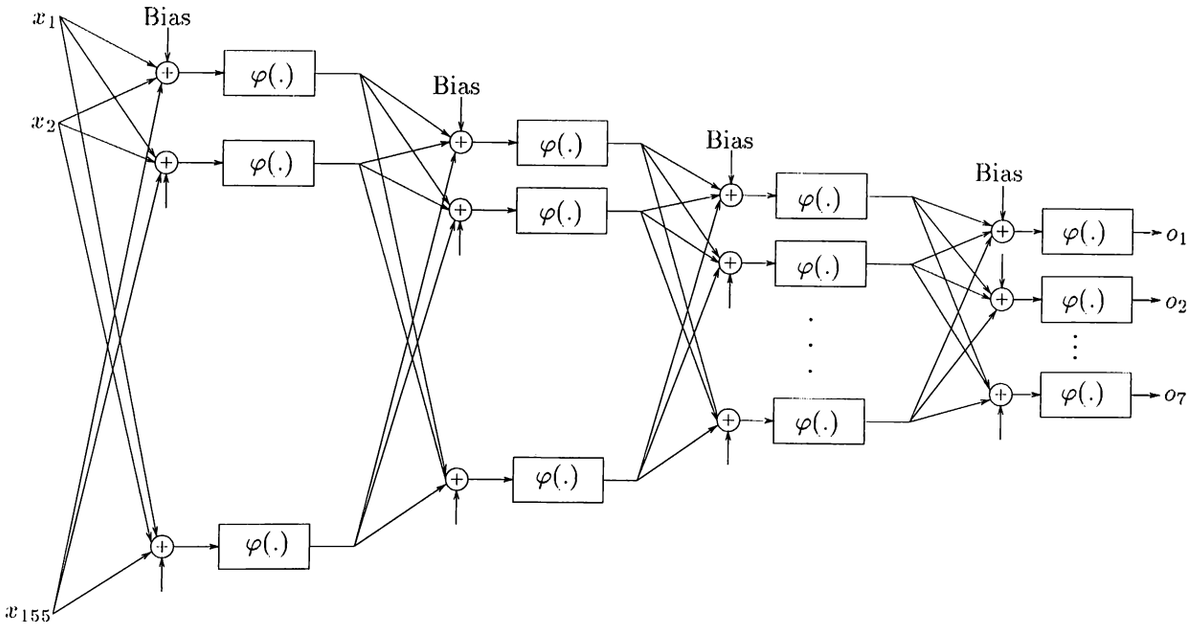


Figure 5.2: Structure of the neural network used in this study.

The input layer of these networks has 155 inputs which are the amplitude and TOF patterns of these target primitives at range r such that:

$$x_i = \begin{cases} A_{aa}(\theta_i) & i = 1, ..., 31 \\ A_{bb}(\theta_{i-31}) & i = 32, ..., 62 \\ A_{ab}(\theta_{i-62}) & i = 63, ..., 93 \\ t_{aa}(\theta_{i-93}) - t_{ab}(\theta_{i-93}) & i = 94, ..., 124 \\ t_{bb}(\theta_{i-124}) - t_{ab}(\theta_{i-124}) & i = 125, ..., 155 \end{cases} \tag{5.3}$$

The number of output layer neurons equals the number of target primitives which is 7 in this study. The output of the network is designed in the training mode as in Table 5.2.

The networks are first trained and then tested. In the testing, the output neuron with the largest value determines the category of the target primitive being classified. The output of the neural networks are found with the help of MATLAB and related programs (Programs 12–13) are given in Appendix F. The nonlinearity used with these networks is the sigmoid function of the form:

$$\varphi(v) = \frac{1}{1 + e^{-v}} \tag{5.4}$$

o_1	o_2	o_3	o_4	o_5	o_6	o_7	DECISION
1	0	0	0	0	0	0	plane
0	1	0	0	0	0	0	corner
0	0	1	0	0	0	0	edge of $\theta_e = 90^o$
0	0	0	1	0	0	0	cylinder with $r_c = 15$ cm
0	0	0	0	1	0	0	acute corner of $\theta_c = 60^o$
0	0	0	0	0	1	0	acute corner of $\theta_c = 45^o$
0	0	0	0	0	0	1	acute corner of $\theta_c = 30^o$

Table 5.2: Desired output patterns of the designed neural network.

While training of the neural network is done along the line-of-sight of the sensor pair in this study, this can be generalized to values of θ other than 0^o by using a polar grid centered at the sensor. This way, isolated targets in front of the sensor pair can be differentiated.

These networks are initially tested by the amplitude and TOF patterns at five different ranges which are 0.5, 1.0, 1.5, 2.0, 2.5 m in a noiseless system. Therefore, each network is tested with 35 test patterns. The success of recognition of these patterns at these ranges for all networks is provided in Tables 5.3–5.8. Referring to these tables, the network can recognize a few target primitives around the range at which the network is trained with the amplitude and TOF patterns. Moreover, it can be concluded that if a corner is learned by a network at range r, this corner can also be recognized by this network at smaller ranges and an interesting result is obtained for an acute corner of $\theta_c = 30^o$ that if an acute corner of $\theta_c = 30^o$ is taught to a network at a range r, it can also be recognized by this network at *larger* ranges. In Table 5.8, only the cylinder of $r_c = 15$ cm cannot be recognized by the network which is taught the amplitude and TOF patterns at five different range values.

r(m)	plane	corner	edge	cylinder	a.c. of $\theta_c = 60^o$	a.c. of $\theta_c = 45^o$	a.c. of $\theta_c = 30^o$
0.5	√	√	√	√	√	√	√
1.0	√	×	×	×	×	×	√
1.5	×	×	×	×	×	×	√
2.0	×	×	×	×	×	×	√
2.5	×	×	×	×	×	×	√

Table 5.3: Success of recognition at different ranges of the neural network which is trained with the patterns at $r = 0.5$ m.

r(m)	plane	corner	edge	cylinder	a.c. of $\theta_c = 60^o$	a.c. of $\theta_c = 45^o$	a.c. of $\theta_c = 30^o$
0.5	×	√	×	×	√	×	×
1.0	√	√	√	√	√	√	√
1.5	√	×	×	√	×	×	√
2.0	×	×	×	×	×	×	√
2.5	×	×	×	×	×	×	√

Table 5.4: Success of recognition at different ranges of the neural network which is trained with the patterns at $r = 1.0$ m.

r(m)	plane	corner	edge	cylinder	a.c. of $\theta_c = 60^o$	a.c. of $\theta_c = 45^o$	a.c. of $\theta_c = 30^o$
0.5	×	√	×	×	×	×	×
1.0	×	√	×	×	√	√	√
1.5	√	√	√	√	√	√	√
2.0	√	×	×	×	√	×	√
2.5	√	×	×	×	×	×	√

Table 5.5: Success of recognition at different ranges of the neural network which is trained with the patterns at $r = 1.5$ m.

r(m)	plane	corner	edge	cylinder	a.c. of $\theta_c = 60^o$	a.c. of $\theta_c = 45^o$	a.c. of $\theta_c = 30^o$
0.5	×	√	×	×	×	×	×
1.0	×	√	×	×	×	×	×
1.5	×	√	×	×	×	×	×
2.0	√	√	√	√	√	√	√
2.5	√	×	×	×	√	×	√

Table 5.6: Success of recognition at different ranges of the neural network which is trained with the patterns at $r = 2.0$ m.

r(m)	plane	corner	edge	cylinder	a.c. of $\theta_c = 60^o$	a.c. of $\theta_c = 45^o$	a.c. of $\theta_c = 30^o$
0.5	×	√	×	×	×	×	×
1.0	×	√	×	×	×	×	×
1.5	×	√	×	×	×	×	×
2.0	×	√	×	×	√	×	√
2.5	√	√	√	√	√	√	√

Table 5.7: Success of recognition at different ranges of the neural network which is trained with the patterns at $r = 2.5$ m.

In the testing mode, a Gaussian white noise with zero mean and standard deviation σ_A is added to the echo amplitude. The TOF of this noisy echo waveform is estimated by thresholding [41]. Then the networks are tested with these noisy amplitude and TOF patterns and the percentages of recognition of these networks for each target primitive and different range values are given in Figures 5.3–5.12. In this testing process, 280 noisy patterns are tested at a single range and 200 noisy patterns are tested for one target primitive. Therefore a total of 1400 noisy patterns are tested for a single amplitude noise standard deviation. In Figures 5.3–5.4, the percentages of success for all target primitives and all range values are constant since echo amplitudes which is reflected from all target primitives at $r = 0.5$ m are very high with respect to the σ_A values used in testing process. For example, the maximum echo amplitude which is reflected from a plane at $r = 0.5$ m is 0.2057 which corresponds to only 4% of maximum value of σ_A used in this testing process. Referring to the rest of these figures, the percentages of success for the first five networks show various trends for each target primitive and different range values except the range at which the network is trained with amplitude and TOF patterns of target primitives. For this range, the percentage of success decreases with increasing noise standard deviation. Since the echo amplitudes decrease with increasing range, the rate of this decrease in the percentage of success also increases with increasing range. For example, while the percentage of success of the network which is trained with the patterns at $r = 0.5$ m is constant at 100% for $r = 0.5$ m (Figure 5.4), it decreases to 66% at $r = 2.5$ m for the network trained with the patterns at $r = 2.5$ m (Figure 5.12). Overall success of these five networks which are the averages of percentage of success for all target primitives at all range values used in this study also show different characteristics in a small interval. For the first network it is constant at 34%, the overall success of the second network is in the interval $[40\%, 43\%]$. Highest overall success is obtained for the third network in the interval $[48\%, 50\%]$ since $r = 1.5$ m is the center of the ranges which are considered in this study. Overall successes for the forth and fifth networks are in the interval $[36\%, 40\%]$ and $[32\%, 38\%]$ respectively.

The testing results of the network which is trained with the amplitude and TOF patterns of target primitives at five different ranges are given in Figures 5.13–5.14. In Figure 5.13, the percentages of success of this network for each target primitive decreases with different rates. It decreases to 39% for an acute corner of $\theta_c = 45^o$ while it decreases to 98% for a plane. This difference comes from the fact that magnitude of echo amplitude reflected from different

r(m)	plane	corner	edge	cylinder	a.c. of $\theta_c = 60^o$	a.c. of $\theta_c = 45^o$	a.c. of $\theta_c = 30^o$
0.5	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$
1.0	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$
1.5	$\surd$	$\surd$	$\surd$	$\times$	$\surd$	$\surd$	$\surd$
2.0	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$
2.5	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$

Table 5.8: Success of recognition at different ranges of the neural network which is trained with the patterns at $r = 0.5, 1.0, 1.5, 2.0$ and 2.5 m.

target primitives are different. Although these noise levels are relatively small compared to echo amplitudes for some of the target primitives, they are very large values for some others like edge or thin cylinder. Since the signal levels are comparable to the noise, it becomes impossible to detect these signals for these target primitives. For example, $\sigma_A = 0.008$ corresponds to 118% of the maximum echo amplitude which is reflected from an edge while it corresponds to only 15% of the maximum echo amplitude which is reflected from a plane at $r = 2.0$ m. In Figure 5.14, the percentages of success of this network at a single range decreases with increasing σ_A and this decrease becomes sharper with increasing range since echo amplitudes which are reflected from all target primitives decreases with increasing range and the signal levels are comparable to the noise at larger ranges.

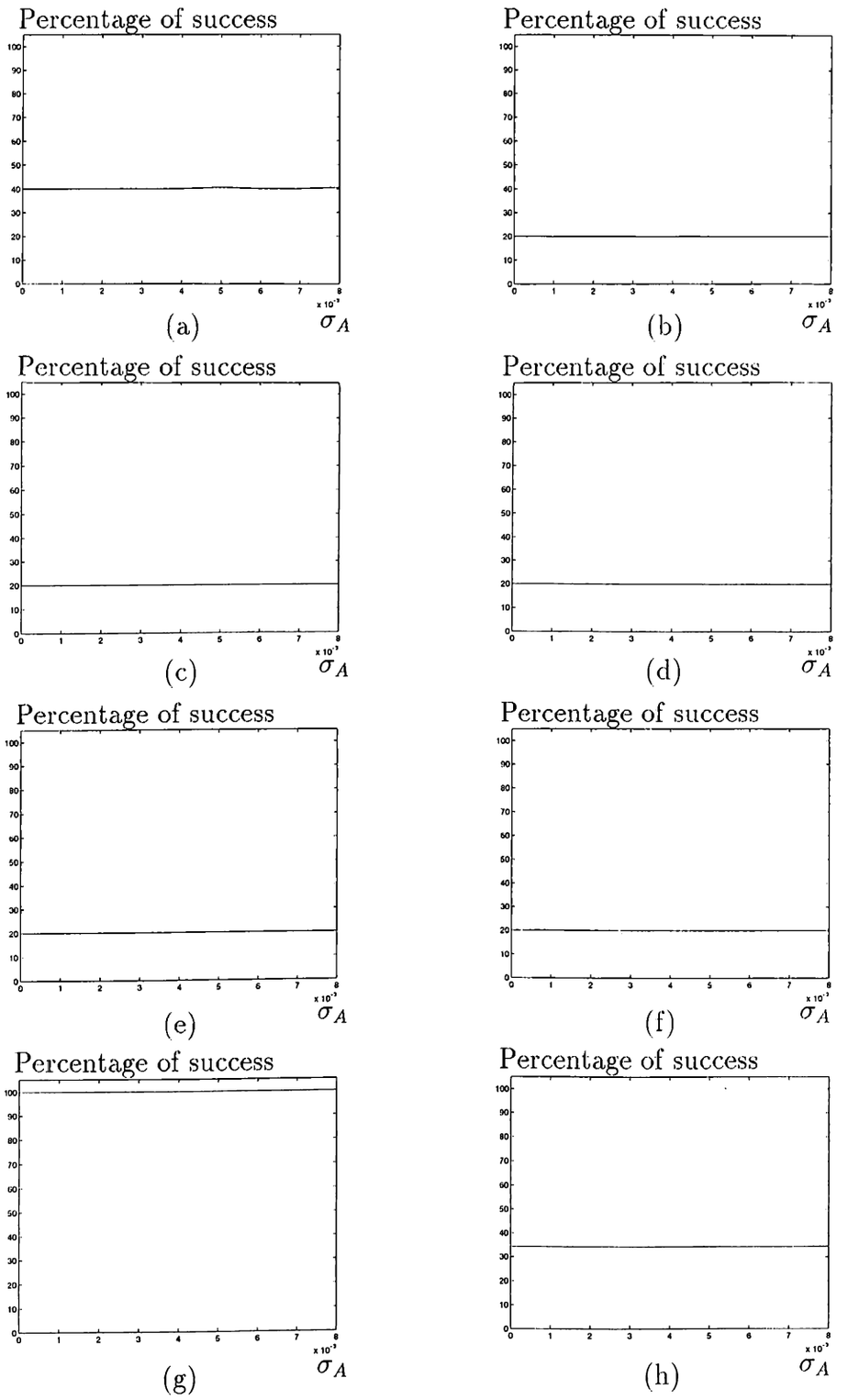


Figure 5.3: Percentage of success of the neural network which is trained with the patterns at $r = 0.5$ m for (a) plane (b) corner (c) edge of $\theta_e = 90^o$ (d) cylinder of $r_c = 15$ cm (e) acute corner of $\theta_c = 60^o$ (f) acute corner of $\theta_c = 45^o$ (g) acute corner of $\theta_c = 30^o$ (h) overall success of this network.

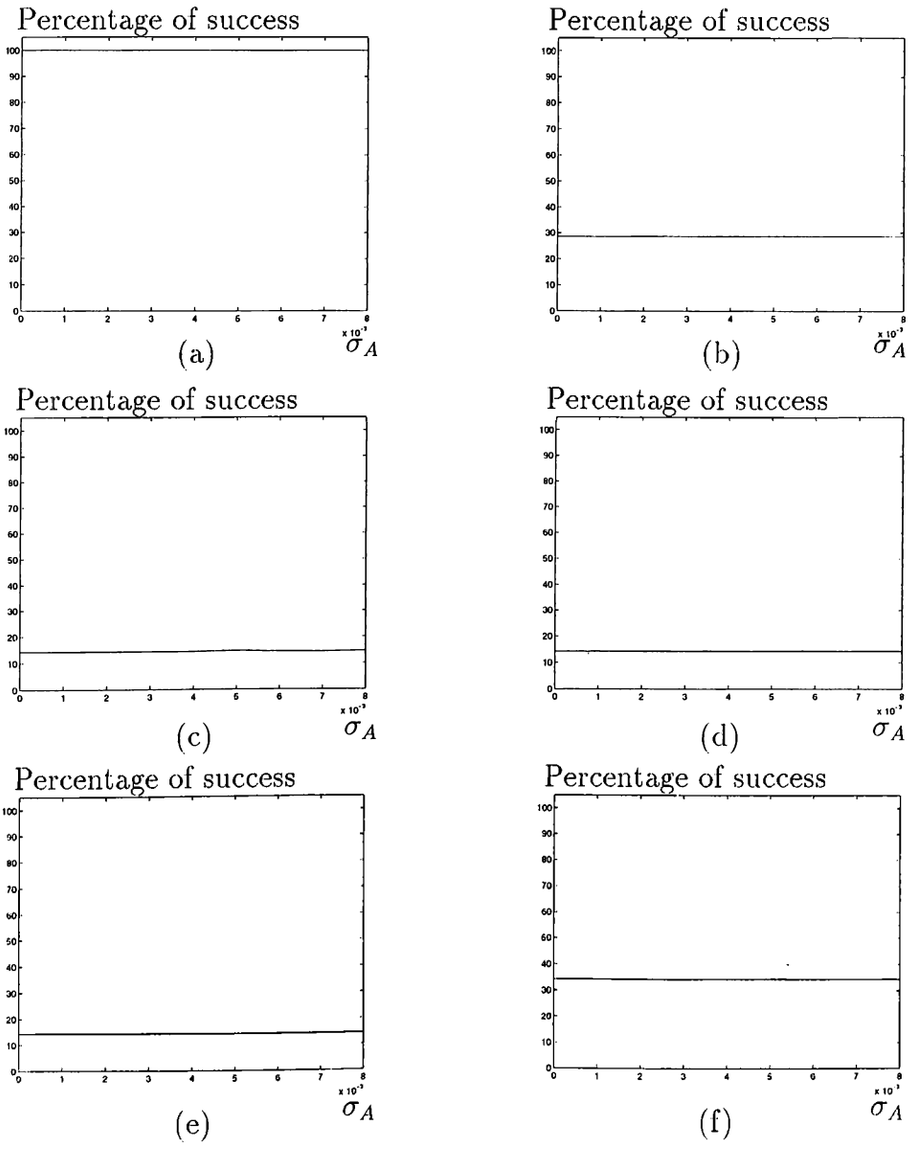


Figure 5.4: Percentage of success of the neural network which is trained with the patterns at $r = 0.5$ m for all target primitives at (a) $r = 0.5$ m (b) $r = 1.0$ m (c) $r = 1.5$ m (d) $r = 2.0$ m (e) $r = 2.5$ m (f) overall success of this network.

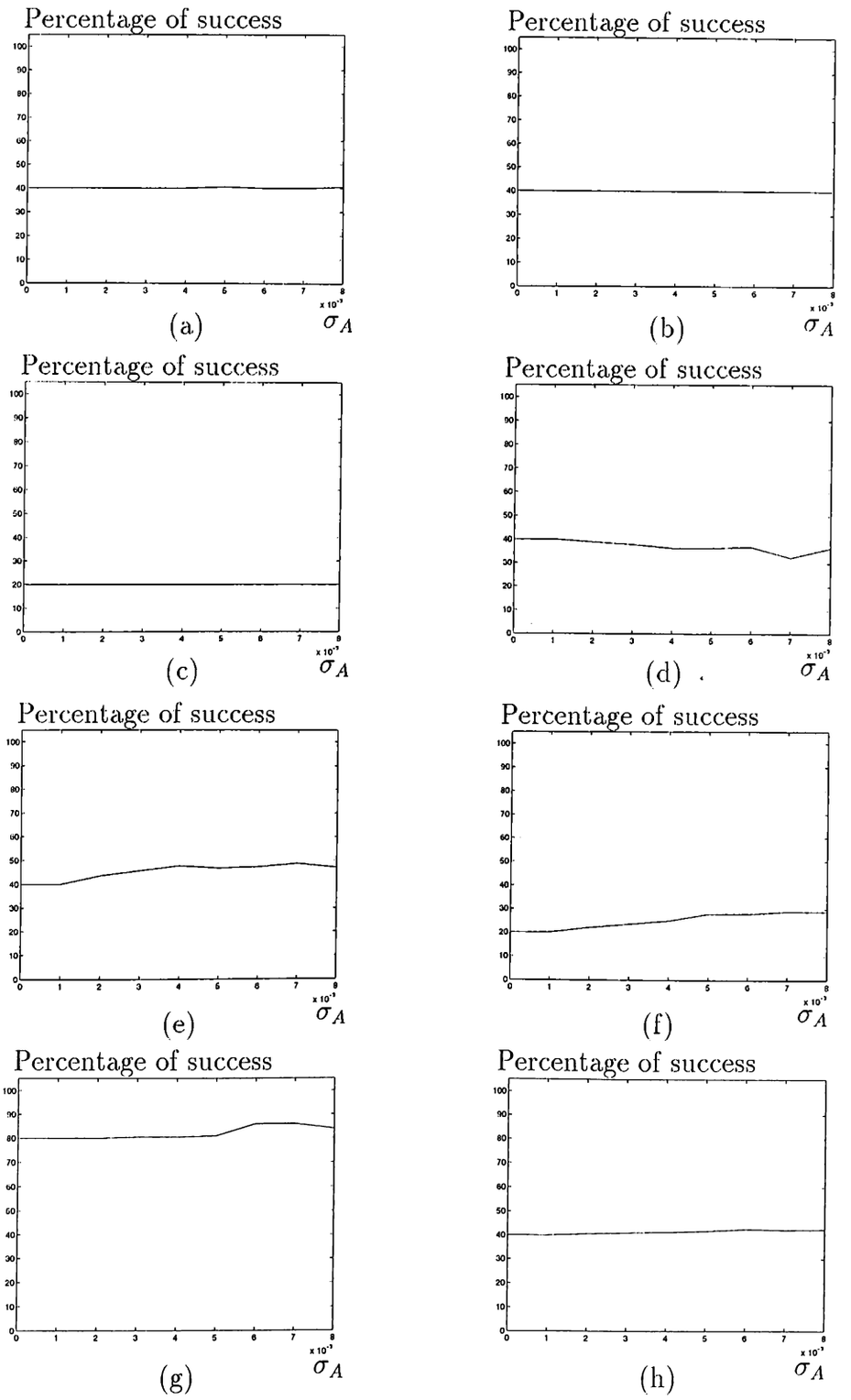


Figure 5.5: Percentage of success of the neural network which is trained with the patterns at $r = 1.0$ m for (a) plane (b) corner (c) edge of $\theta_e = 90^o$ (d) cylinder of $r_c = 15$ cm (e) acute corner of $\theta_c = 60^o$ (f) acute corner of $\theta_c = 45^o$ (g) acute corner of $\theta_c = 30^o$ (h) overall success of this network.

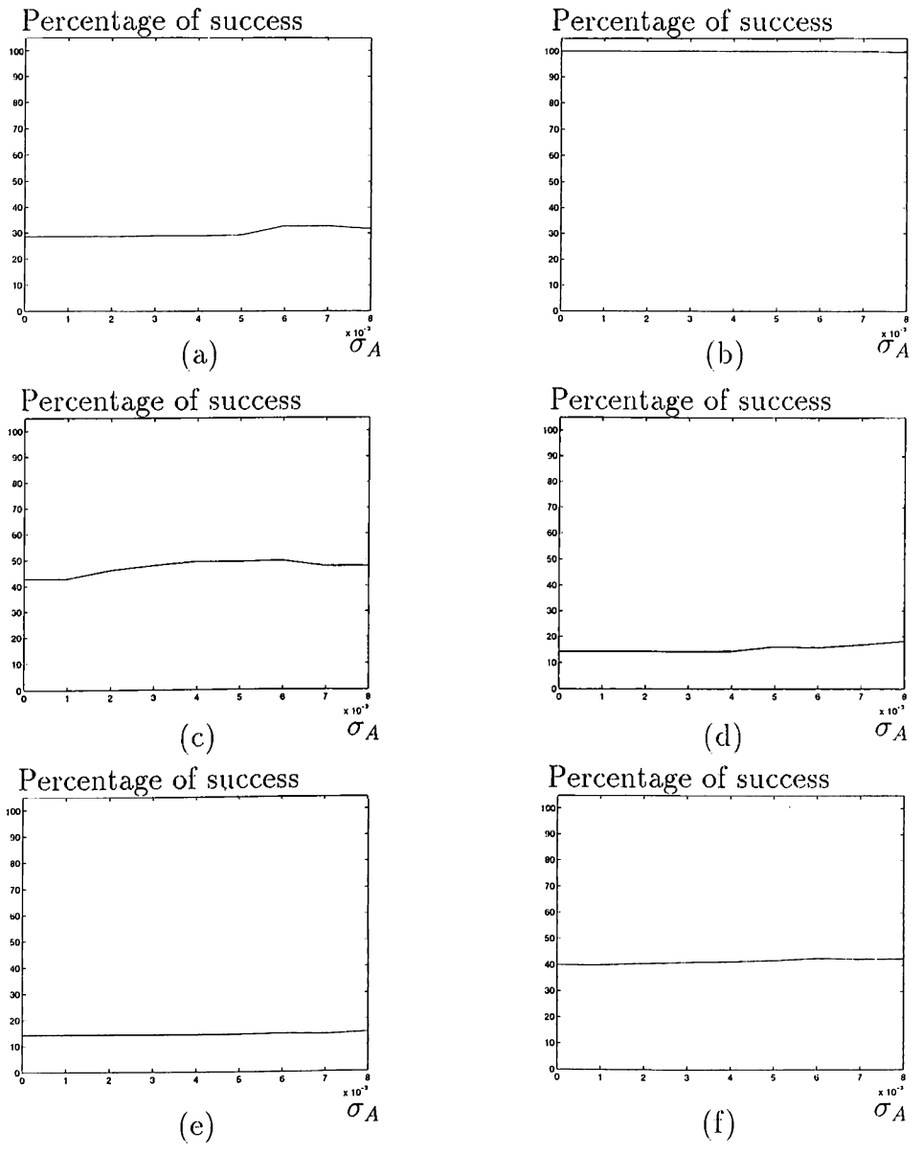


Figure 5.6: Percentage of success of the neural network which is trained with the patterns at $r = 1.0$ m for all target primitives at (a) $r = 0.5$ m (b) $r = 1.0$ m (c) $r = 1.5$ m (d) $r = 2.0$ m (e) $r = 2.5$ m (f) overall success of this network.

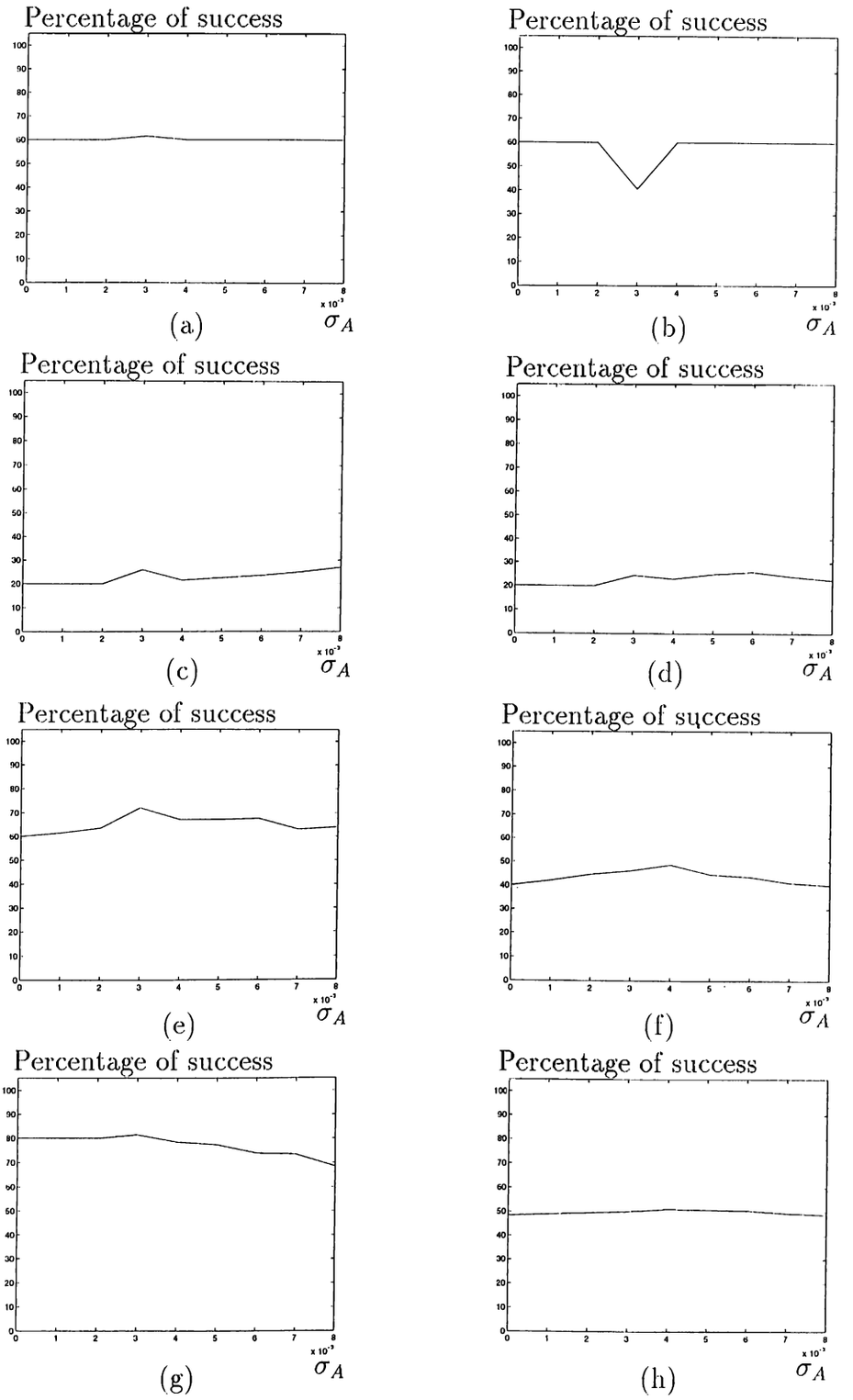


Figure 5.7: Percentage of success of the neural network which is trained with the patterns at $r = 1.5$ m for (a) plane (b) corner (c) edge of $\theta_e = 90^o$ (d) cylinder of $r_c = 15$ cm (e) acute corner of $\theta_c = 60^o$ (f) acute corner of $\theta_c = 45^o$ (g) acute corner of $\theta_c = 30^o$ (h) overall success of this network.

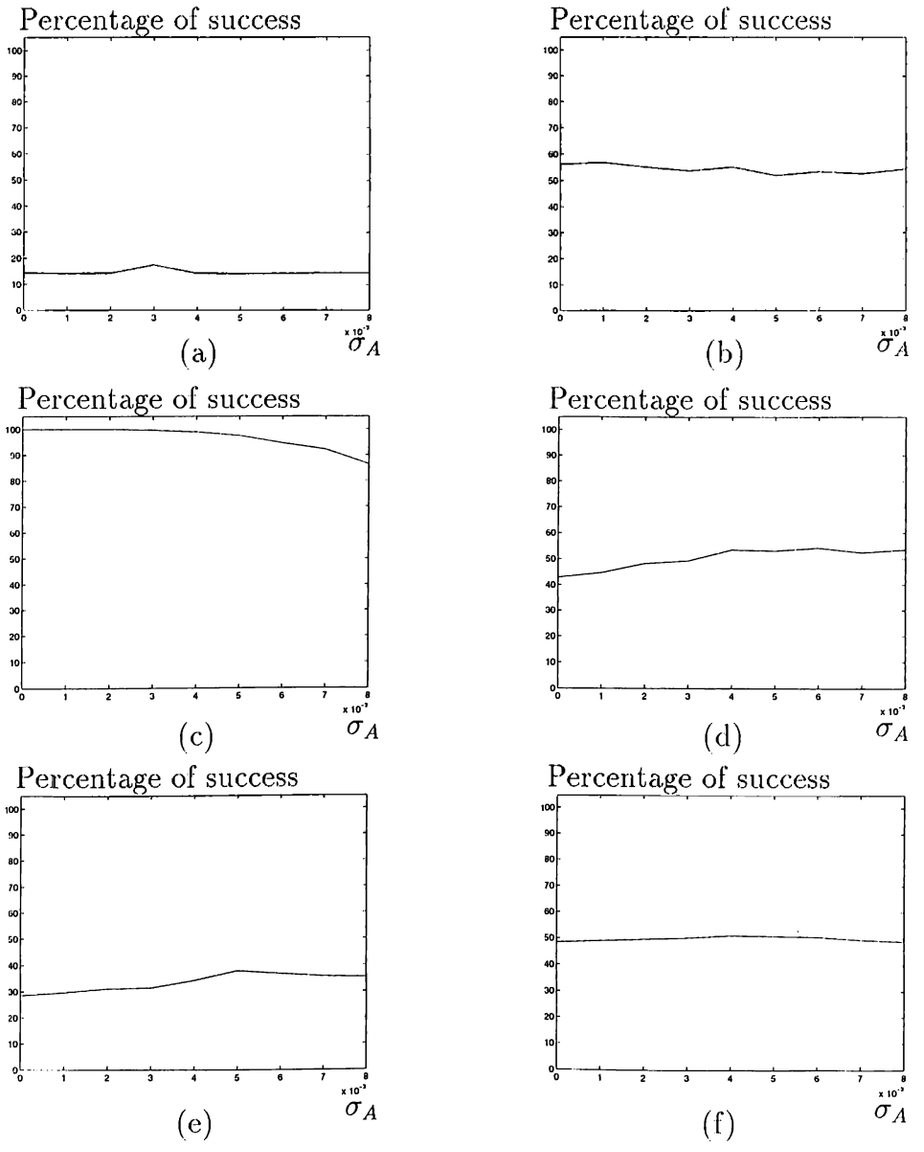


Figure 5.8: Percentage of success of the neural network which is learned with the patterns at $r = 1.5$ m for all target primitives at (a) $r = 0.5$ m (b) $r = 1.0$ m (c) $r = 1.5$ m (d) $r = 2.0$ m (e) $r = 2.5$ m (f) overall success of this network.

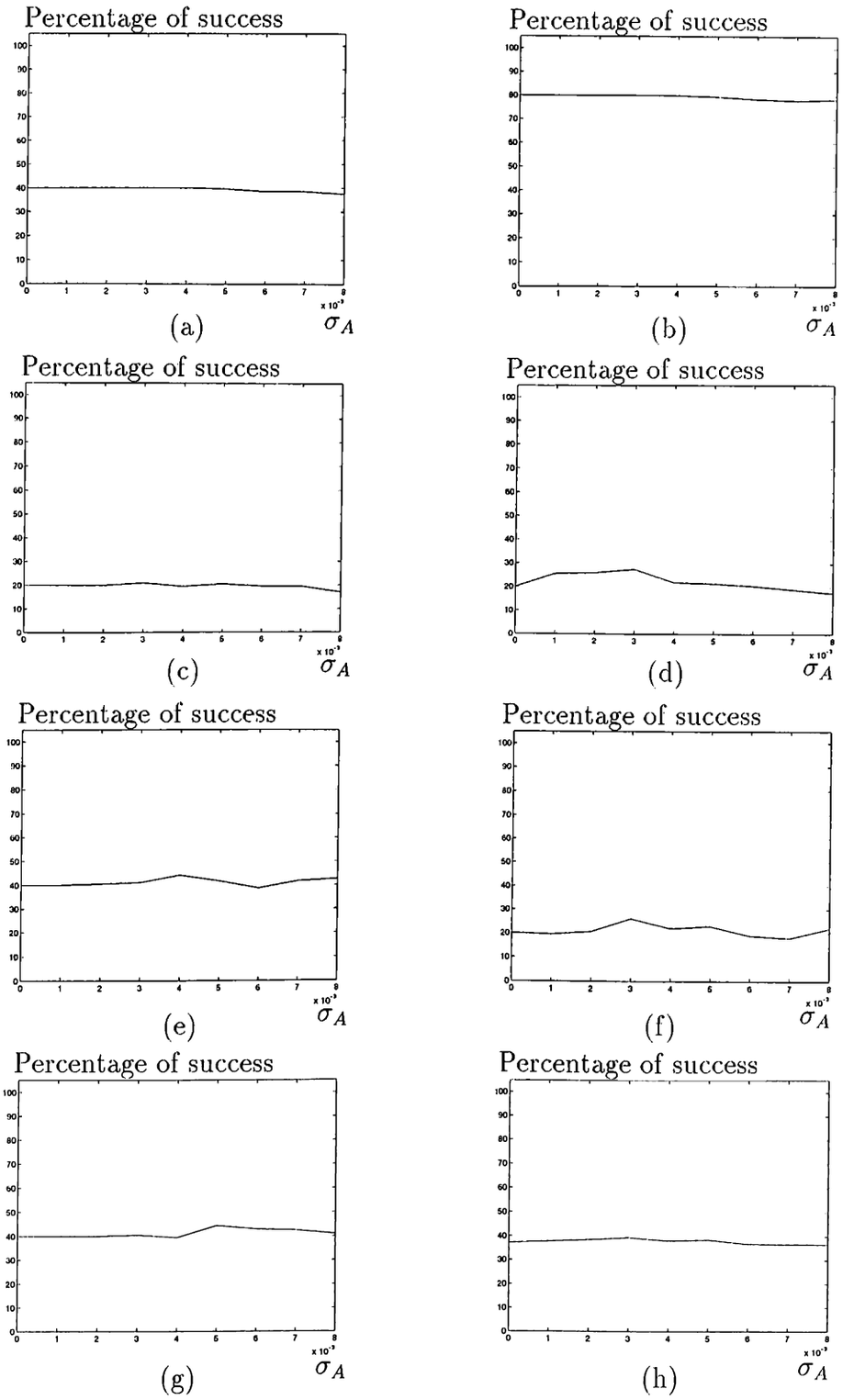


Figure 5.9: Percentage of success of the neural network which is trained with the patterns at $r = 2.0$ m for (a) plane (b) corner (c) edge of $\theta_e = 90^o$ (d) cylinder of $r_c = 15$ cm (e) acute corner of $\theta_c = 60^o$ (f) acute corner of $\theta_c = 45^o$ (g) acute corner of $\theta_c = 30^o$ (h) overall success of this network.

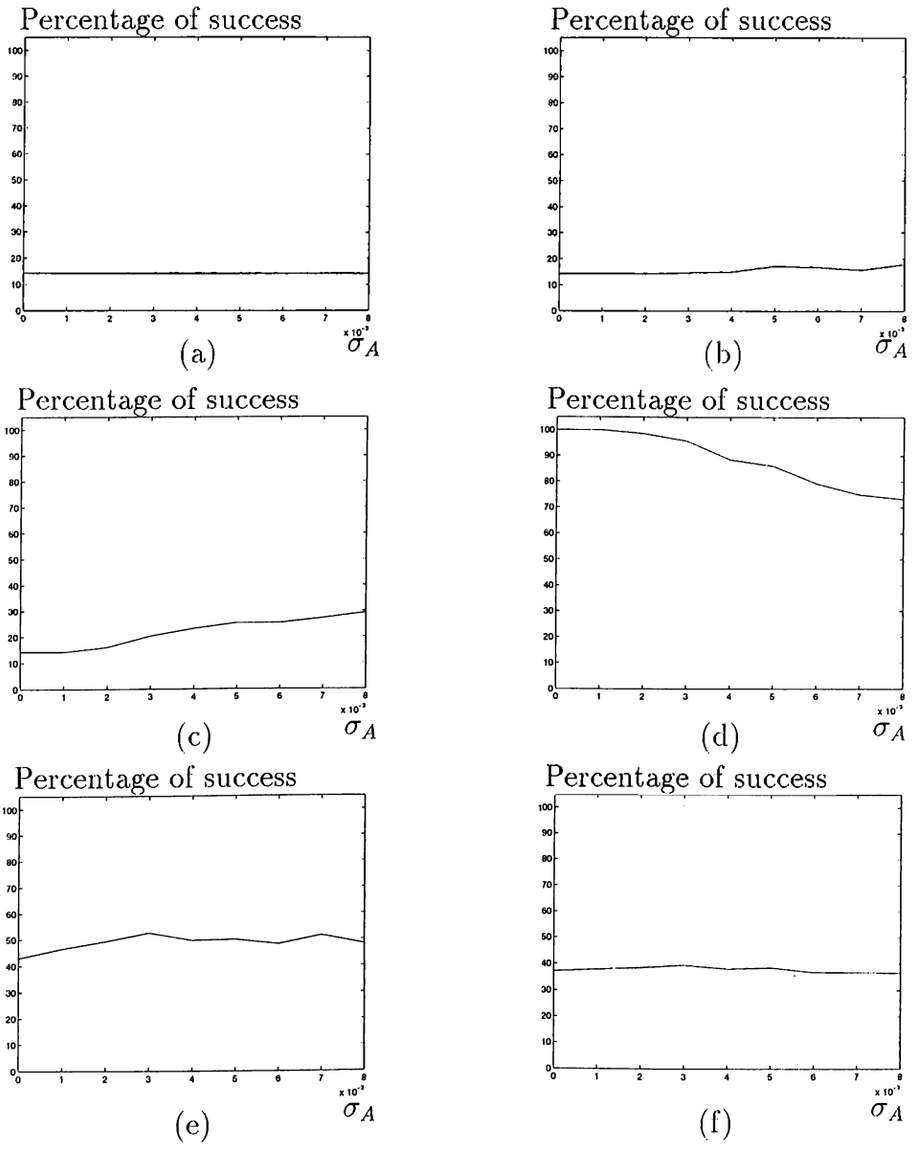


Figure 5.10: Percentage of success of the neural network which is trained with the patterns at $r = 2.0$ m for all target primitives at (a) $r = 0.5$ m (b) $r = 1.0$ m (c) $r = 1.5$ m (d) $r = 2.0$ m (e) $r = 2.5$ m (f) overall success of this network.

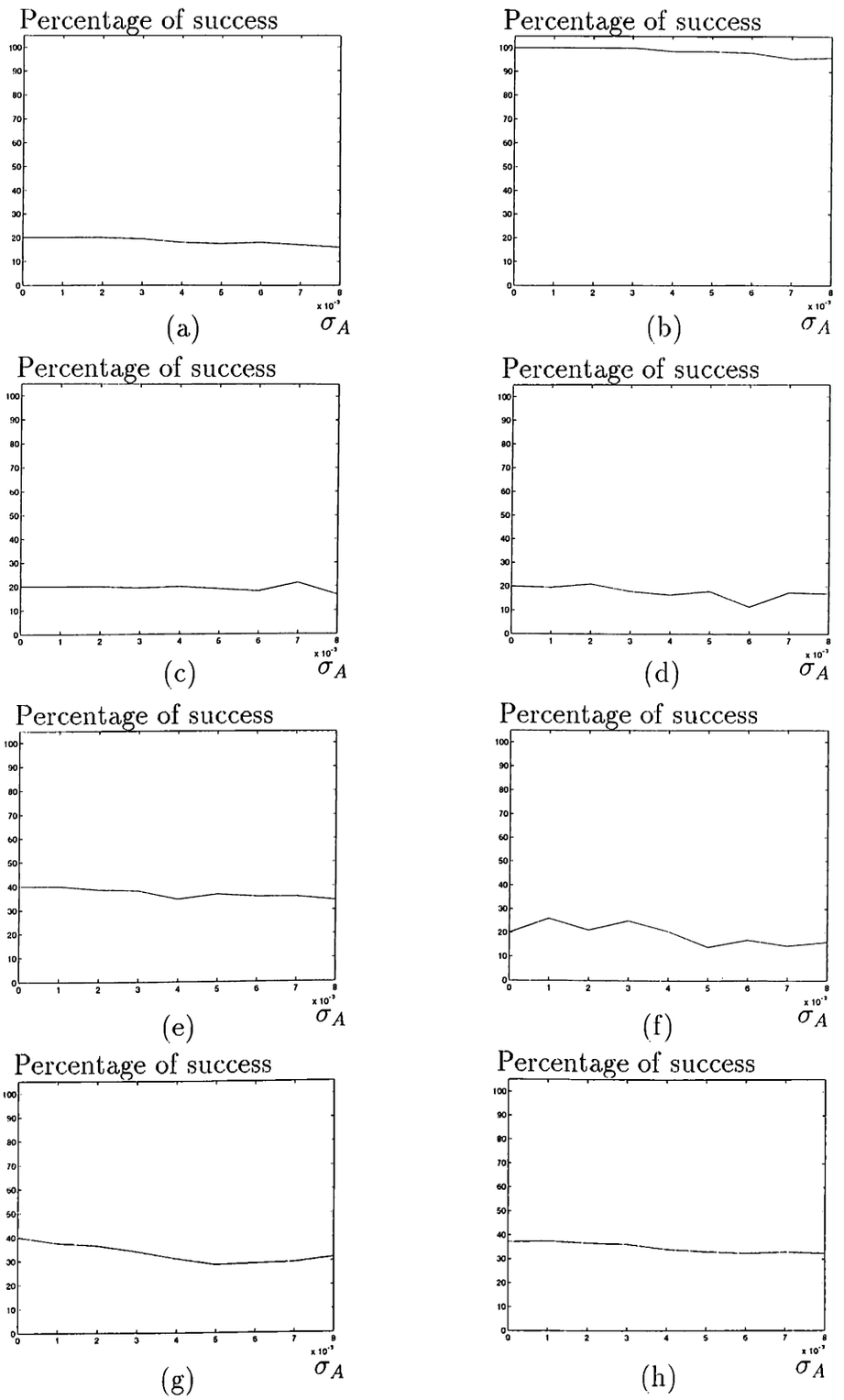


Figure 5.11: Percentage of success of the neural network which is trained with the patterns at $r = 2.5$ m for (a) plane (b) corner (c) edge of $\theta_e = 90^o$ (d) cylinder of $r_c = 15$ cm (e) acute corner of $\theta_c = 60^o$ (f) acute corner of $\theta_c = 45^o$ (g) acute corner of $\theta_c = 30^o$ (h) overall success of this network.

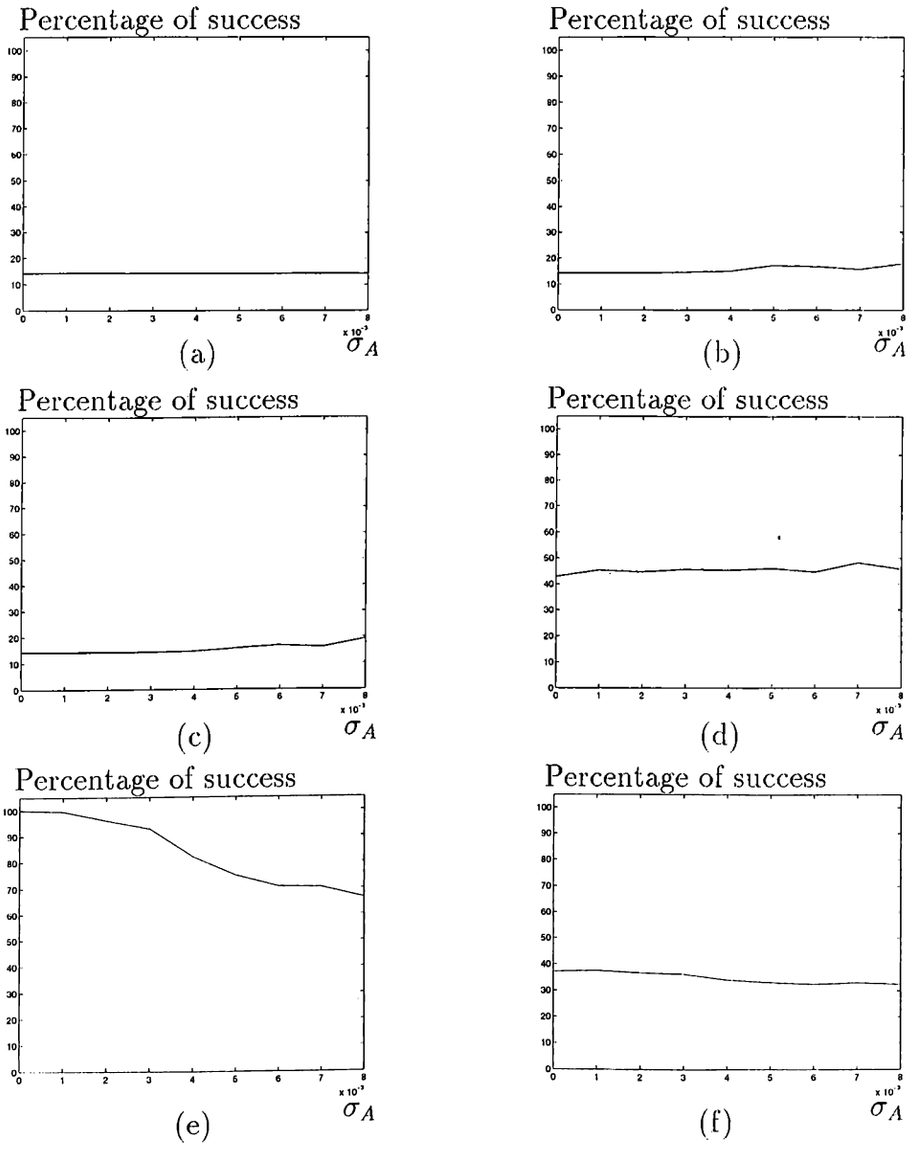


Figure 5.12: Percentage of success of the neural network which is trained with the patterns at $r = 2.5$ m for all target primitives at (a) $r = 0.5$ m (b) $r = 1.0$ m (c) $r = 1.5$ m (d) $r = 2.0$ m (e) $r = 2.5$ m (f) overall success of this network.

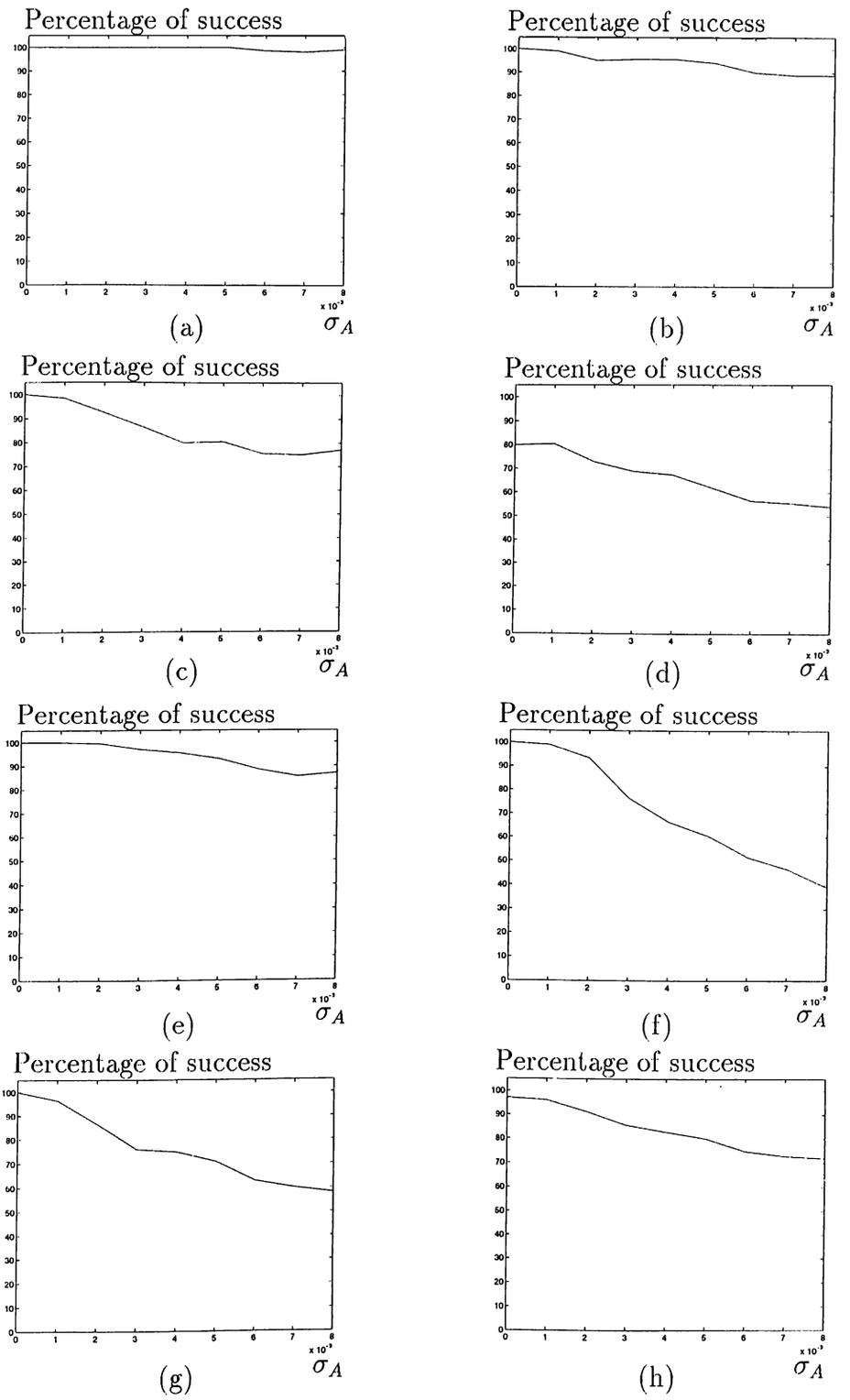


Figure 5.13: Percentage of success of the neural network which is trained with the patterns at $r = 0.5, 1.0, 1.5, 2.0$ and 2.5 m for (a) plane (b) corner (c) edge of $\theta_e = 90^o$ (d) cylinder of $r_c = 15$ cm (e) acute corner of $\theta_c = 60^o$ (f) acute corner of $\theta_c = 45^o$ (g) acute corner of $\theta_c = 30^o$ (h) overall success of this network.

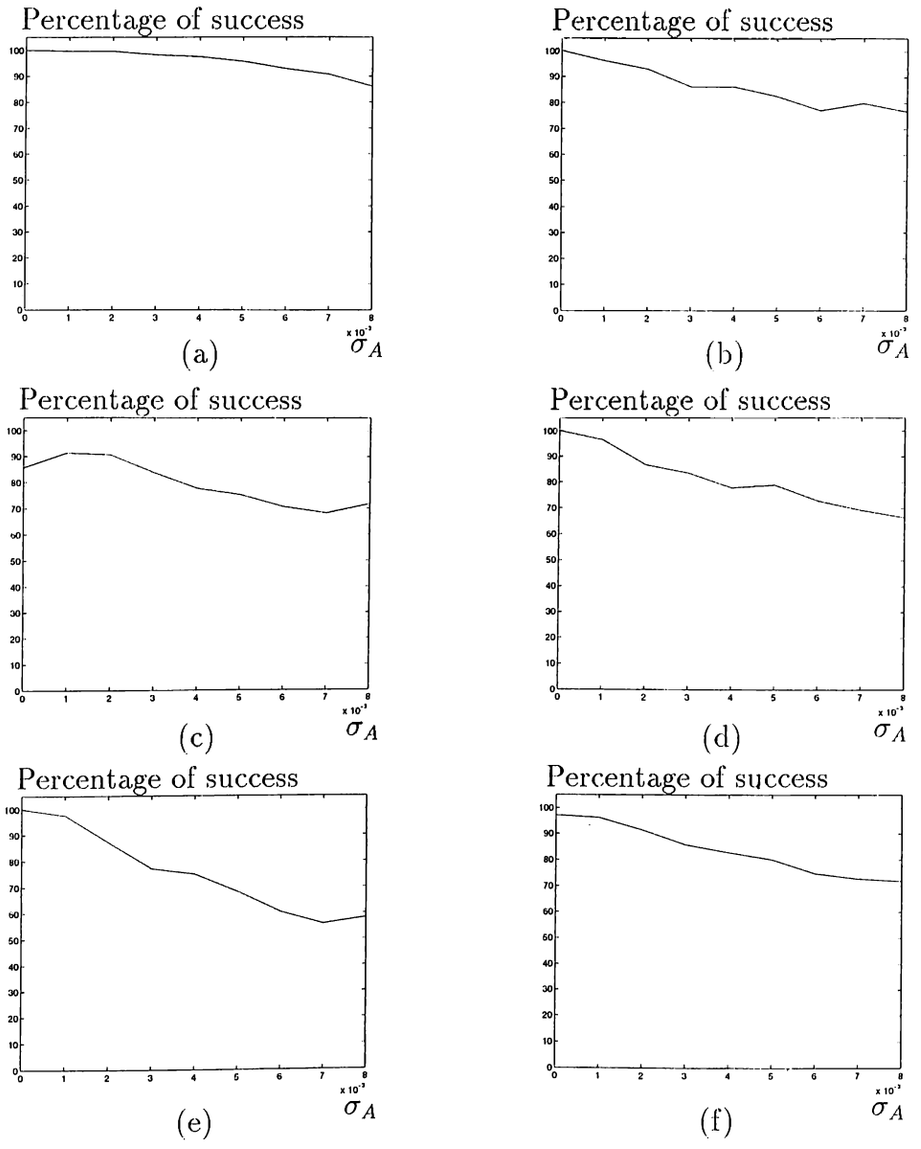


Figure 5.14: Percentage of success of the neural network which is trained with the patterns at $r = 0.5, 1.0, 1.5, 2.0$ and 2.5 m for all target primitives at (a) $r = 0.5$ m (b) $r = 1.0$ m (c) $r = 1.5$ m (d) $r = 2.0$ m (e) $r = 2.5$ m (f) overall success of this network.

In this chapter neural network classifiers are designed to identify the target primitives which are: plane, corner, edge of $\theta_e = 90^o$, cylinder of $r_c = 15$ cm and acute corners of $\theta_c = 30^o$, $\theta_c = 45^o$ and $\theta_c = 60^o$. Testing results of these networks for each target primitive and each range values are also provided. In the next chapter, discussion and concluding remarks are given and directions for future research are motivated.

Chapter 6

CONCLUSION

6.1 Discussion

The neural network classifiers used in Chapter 4 are non-parametric and do not make any assumptions on the underlying distribution of input data. The belief function approach which is employed in the differentiation of plane, corner and acute corner targets also enables the modeling of non-parametric uncertainty but it assumes echo amplitude noise as Gaussian while assigning beliefs to these target primitives. Although networks were trained for fixed θ in this study, the tramp can be generalized to a polar grid. This way, targets at arbitrary locations with the sensitivity pattern of the sensor pair can be discriminated and localized.

In the belief function case, only plane, corner and acute corner can be differentiated by using the measurements of a single sensor pair. However in the neural network case, all target primitives which are considered in this study can be differentiated. Although 31 measurements obtained from a sonar scan of a single sensor pair were processed here, the processing time could be reduced by taking fewer measurements.

Employing Dempster-Shafer rule of combination to improve the performance of classification by reducing perception uncertainty in the feature fusion from three sensor pairs, improvement in differentiation is 20% without false decision, at the cost of additional computation. However the overall performance of the network which is designed by training the amplitude and TOF patterns of target primitives at five different ranges decreases only to 72% at

$\sigma_A = 0.008$ with 28% false decision rate. This is of course at the cost of elapsed time until the network learns these patterns.

As a result, one cannot say one method is superior to another since it depends on application. For example, if you are working in a well structured environment, neural network approach can be used efficiently. However, if the working environment is unstructured, then belief function approach can be employed efficiently.

6.2 Conclusion

In this study, physical models are used to model reflections from target primitives commonly encountered in mobile robot applications. Target features are generated as being evidentially tied to degrees of belief which are subsequently fused for multiple sonars at different geographical sites. Using both TOF and amplitude data in the feature fusion allows more robust differentiation. As a first method, the belief function approach employed in the differentiation of plane, corner and acute corner enables the modeling of non-parametric uncertainty. Fusion of feature data from multiple sensors using Dempster-Shafer rule of combination reduces such perception uncertainty but increases the processing time by approximately twofold. In the second approach, non-parametric neural network classifiers are employed which do not make any assumptions on the underlying distribution of input data for an incorporating learning of identifying parameter relations of target primitives used in this study.

As for future work, the sensors can be extended to include infrared and laser systems in the data fusion process for target identification and target localization. Moreover, target identification can be expanded to 3-D. The number of target primitives can be increased in 3-D by considering the position and the rotation of planes which form 3-D corners and edges.

Appendix A

GEOMETRIC TARGET AND ECHO SIGNAL MODELS

In this appendix, geometric models of target primitives used in this study are provided. The targets modeled are plane, corner, acute corner, edge and cylinder. For each target primitive, models of the corresponding echo signals are given which are based on experimental data.

A.1 Planar Target Model

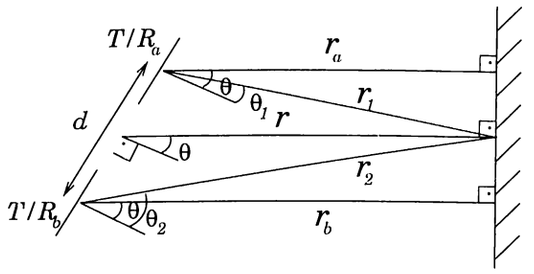


Figure A.1: Geometry of the problem with the given sensor pair when the target is a plane.

A *plane* has a line segment profile in 2-D. The geometry of reflection for planar target is illustrated in Figure A.1. From this geometry, $r_1, r_2, r_a, r_b, \theta_1$ and θ_2 can be expressed as:

$$r_1 = \sqrt{r^2 + \frac{d^2}{4} - dr\sin\theta} \tag{A.1}$$

$$r_2 = \sqrt{r^2 + \frac{d^2}{4} + dr\sin\theta} \tag{A.2}$$

$$r_a = r - \frac{d}{2}\sin\theta \tag{A.3}$$

$$r_b = r + \frac{d}{2}\sin\theta \tag{A.4}$$

$$\theta_1 = \cos^{-1}\left(\frac{r\cos\theta}{r_1}\right) \tag{A.5}$$

$$\theta_2 = \cos^{-1}\left(\frac{r\cos\theta}{r_2}\right) \tag{A.6}$$

Since each transducer can be employed both as transmitter and receiver, a set of four TOF and four amplitude measurements are obtained each time data is collected from the target. From the geometry, the TOF measurements for each transmitter and receiver pair are found as:

$$t_{aa} = \frac{2r_a}{c} \tag{A.7}$$

$$t_{ab} = t_{ba} = \frac{r_1 + r_2}{c} \tag{A.8}$$

$$t_{bb} = \frac{2r_b}{c} \tag{A.9}$$

where t_{ab} denotes TOF extracted from $A_{ab}(r,\theta,d,t)$. $A_{ab}(r,\theta,d,t)$ denotes the signal which is transmitted by a and received by b at time t. The detected signals by each transmitter and receiver pair can be modeled as:

$$A_{aa}(r,\theta,d,t) = A_{max}\frac{r_{min}}{r_a}e^{-\frac{\theta^2}{\sigma_T^2}}e^{-\frac{(t-t_{aa}-\frac{3}{f_o})^2}{\sigma^2}}\sin[2\pi f_0(t-t_{aa})] \tag{A.10}$$

$$A_{ab}(r,\theta,d,t) = A_{max}\frac{2r_{min}}{r_1+r_2}e^{-\frac{\theta_1^2+\theta_2^2}{2\sigma_T^2}}e^{-\frac{(t-t_{ab}-\frac{3}{f_o})^2}{\sigma^2}}\sin[2\pi f_0(t-t_{ab})] \tag{A.11}$$

$$A_{ba}(r,\theta,d,t) = A_{max}\frac{2r_{min}}{r_2+r_1}e^{-\frac{\theta_2^2+\theta_1^2}{2\sigma_T^2}}e^{-\frac{(t-t_{ba}-\frac{3}{f_o})^2}{\sigma^2}}\sin[2\pi f_0(t-t_{ba})] \tag{A.12}$$

$$A_{bb}(r,\theta,d,t) = A_{max}\frac{r_{min}}{r_b}e^{-\frac{\theta^2}{\sigma_T^2}}e^{-\frac{(t-t_{bb}-\frac{3}{f_o})^2}{\sigma^2}}\sin[2\pi f_0(t-t_{bb})] \tag{A.13}$$

The form of the range attenuation term $r_1 + r_2$ in (A.11−A.12) is due to specular reflection of the beam. Since r_{min} is the minimum distance at which beam pattern model described above is valid, the beam will be in the near zone when $r < r_{min}$ where the far-zone beam model cannot be used anymore. According to the piston model of the transducer [26], in the near zone, the beam is confined in a cylinder with radius a which is the radius of the transducer. The length of this zone is approximately $\frac{a^2}{\lambda}$ from the face of transducer where λ is the wavelength of the acoustic transmission. On the other hand, the beam is confined in a cone with angle $2\theta_o$ in the far zone.

A.2 Corner Target Model

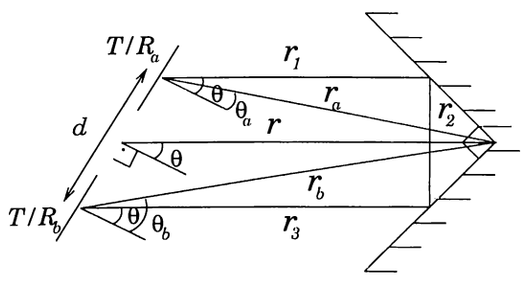


Figure A.2: Geometry of the problem when the target is a corner.

A *right-angle corner* is the line between two perpendicular planes that form a concave dihedral. When the target is a corner, the received signal consists of two components: First component is the diffracted signal which originates from the junction of the two planes, the second is the doubly reflected signal from the two intersecting planes. The diffracted signal can be neglected since its amplitude contribution is much smaller than that of the reflected signal. From the geometry of Figure A.2, r_a, r_b, θ_a and θ_b can be derived as:

$$r_a = \sqrt{r^2 + \frac{d^2}{4} - dr\sin\theta} \tag{A.14}$$

$$r_b = \sqrt{r^2 + \frac{d^2}{4} + dr\sin\theta} \tag{A.15}$$

$$\theta_a = \cos^{-1}\left(\frac{r\cos\theta}{r_a}\right) \tag{A.16}$$

$$\theta_b = \cos^{-1}\left(\frac{r\cos\theta}{r_b}\right) \tag{A.17}$$

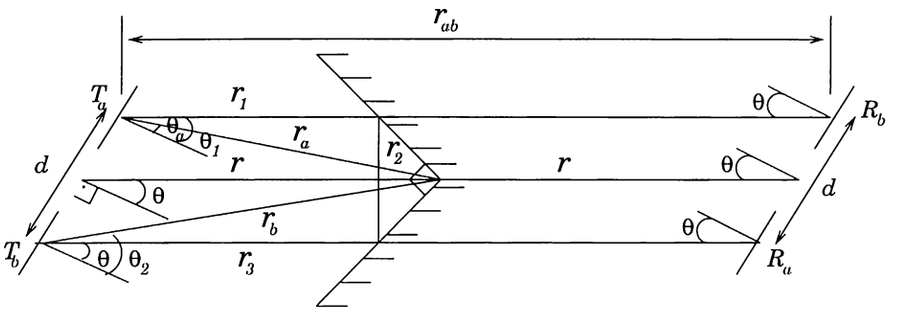


Figure A.3: Geometry of the problem when each surface reflects specularly.

However, to find r_{ab} which denotes the distance that the beam travels from transmitter a to receiver b which equals $r_1 + r_2 + r_3$ in Figure A.2, each plane that constitutes the corner is considered as a mirror reflector in Figure A.3. Due to the configuration of virtual receivers, r_{ab} and r_{ba} are both equal to twice the range:

$$r_{ab} = r_{ba} = 2r \tag{A.18}$$

Therefore, the TOF measurements for each transmitter can be found as:

$$t_{aa} = \frac{2r_a}{c} \tag{A.19}$$

$$t_{ab} = t_{ba} = \frac{2r}{c} \tag{A.20}$$

$$t_{bb} = \frac{2r_b}{c} \tag{A.21}$$

The sonar signal reflected from a corner target can be modeled as:

$$A_{aa}(r,\theta,d,t) = A_{max}\frac{r_{min}}{r_a}e^{-\frac{\theta_a^2}{\sigma_T^2}}e^{-\frac{(t-t_{aa}-\frac{3}{f_o})^2}{\sigma^2}}\sin[2\pi f_0(t-t_{aa})] \tag{A.22}$$

$$A_{ab}(r,\theta,d,t) = A_{max}\frac{r_{min}}{r}e^{-\frac{\theta^2}{\sigma_T^2}}e^{-\frac{(t-t_{ab}-\frac{3}{f_o})^2}{\sigma^2}}\sin[2\pi f_0(t-t_{ab})] \tag{A.23}$$

$$A_{ba}(r,\theta,d,t) = A_{max}\frac{r_{min}}{r}e^{-\frac{\theta^2}{\sigma_T^2}}e^{-\frac{(t-t_{ba}-\frac{3}{f_o})^2}{\sigma^2}}\sin[2\pi f_0(t-t_{ba})] \tag{A.24}$$

$$A_{bb}(r,\theta,d,t) = A_{max}\frac{r_{min}}{r_b}e^{-\frac{\theta_b^2}{\sigma_T^2}}e^{-\frac{(t-t_{bb}-\frac{3}{f_o})^2}{\sigma^2}}\sin[2\pi f_0(t-t_{bb})] \tag{A.25}$$

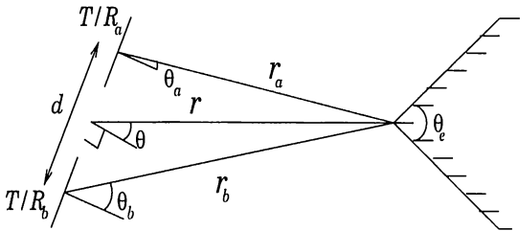


Figure A.4: Geometry of the problem when the target is an edge.

A.3 Edge Target Model

An *edge* is the intersection of two planes that form a convex space when observed from the transducers. For the edge, the acoustic signal diffracts and spreads out cylindrically once it reaches the line defining the edge. Due to the relatively lower echo amplitude, edges are the most difficult and problematic targets to detect in a mobile robot's environment. The expressions for r_a, r_b, θ_a and θ_b are same as those for the corner given in Equations A.14–A.17 respectively. For the edge, the TOF measurements of each transmitter and receiver pair can be found as:

$$t_{aa} = \frac{2r_a}{c} \tag{A.26}$$

$$t_{ab} = t_{ba} = \frac{r_a + r_b}{c} \tag{A.27}$$

$$t_{bb} = \frac{2r_b}{c} \tag{A.28}$$

The signal waveforms are modeled as:

$$A_{aa}(r,\theta,d,t) = \rho_e \frac{A_{max} r_{min}^{3/2}}{r_a^{3/2}} e^{-\frac{\theta_a^2}{\sigma_T^2}} e^{-\frac{(t-t_{aa}-\frac{3}{f_o})^2}{\sigma^2}} \sin[2\pi f_0(t-t_{aa})] \tag{A.29}$$

$$A_{ab}(r,\theta,d,t) = \rho_e \frac{A_{max} r_{min}^{3/2}}{r_a r_b^{1/2}} e^{-\frac{\theta_a^2+\theta_b^2}{2\sigma_T^2}} e^{-\frac{(t-t_{ab}-\frac{3}{f_o})^2}{\sigma^2}} \sin[2\pi f_0(t-t_{ab})] \tag{A.30}$$

$$A_{ba}(r,\theta,d,t) = \rho_e \frac{A_{max} r_{min}^{3/2}}{r_b r_a^{1/2}} e^{-\frac{\theta_b^2+\theta_a^2}{2\sigma_T^2}} e^{-\frac{(t-t_{ba}-\frac{3}{f_o})^2}{\sigma^2}} \sin[2\pi f_0(t-t_{ba})] \tag{A.31}$$

$$A_{bb}(r,\theta,d,t) = \rho_e \frac{A_{max} r_{min}^{3/2}}{r_b^{3/2}} e^{-\frac{\theta_b^2}{\sigma_T^2}} e^{-\frac{(t-t_{bb}-\frac{2}{f_0})^2}{\sigma^2}} \sin[2\pi f_0(t-t_{bb})] \qquad \text{(A.32)}$$

where ρ_e is a reflection coefficient which decreases with θ_e, defining the sharpness of the wedge. Based on experimental data, ρ_e can be approximately modeled as a linear function of θ_e which is the angle of the edge ($\theta_e \leq \frac{\pi}{2}$) given in Figure A.4 [42] such that:

$$\rho_e \frac{A_{max} r_{min}^{3/2}}{r_a} = 0.16 \quad \text{if} \quad \theta_e = 90^o \qquad \text{(A.33)}$$

and

$$\rho_e \frac{A_{max} r_{min}^{3/2}}{r_a} = 0.08 \quad \text{if} \quad \theta_e = 60^o \qquad \text{(A.34)}$$

where $A_{max} = 1$, $r_{min} = 10$ cm and $r_a \cong 10$ cm. Then, ρ_e can be found as:

$$\rho_e = 8.43 \times 10^{-3} \theta_e - 0.25298 \qquad \text{(A.35)}$$

where θ_e is in degrees.

A.4 Cylinder Target Model

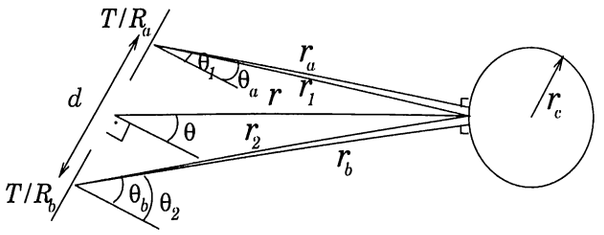


Figure A.5: Geometry of the problem when the target is a cylinder with radius r_c.

Although not as common as planes and corners, a mobile robot very often comes across a cylindrical target in its environment in the form of pillars or poles. Experimental observations indicate that for a cylinder with radius 2.54 cm located at 10 cm, the echo amplitude is observed to be 50% of that from a normally incident plane at the same range [42]. From the geometry of Figure A.5, $r_1, r_2, r_a, r_b, \theta_1, \theta_2, \theta_a$ and θ_b are found as:

$$r_1 = \sqrt{r^2 + \frac{d^2}{4} - dr\sin\theta} \qquad \text{(A.36)}$$

$$r_2 = \sqrt{r^2 + \frac{d^2}{4} + dr\sin\theta} \tag{A.37}$$

$$r_a = \sqrt{(r + r_c)^2 + \frac{d^2}{4} - d(r + r_c)\sin\theta} - r_c \tag{A.38}$$

$$r_b = \sqrt{(r + r_c)^2 + \frac{d^2}{4} + d(r + r_c)\sin\theta} - r_c \tag{A.39}$$

$$\theta_1 = \cos^{-1}\left(\frac{r\cos\theta}{r_1}\right) \tag{A.40}$$

$$\theta_2 = \cos^{-1}\left(\frac{r\cos\theta}{r_2}\right) \tag{A.41}$$

$$\theta_a = \cos^{-1}\left[\frac{(r + r_c)\cos\theta}{(r_a + r_c)}\right] \tag{A.42}$$

$$\theta_b = \cos^{-1}\left[\frac{(r + r_c)\cos\theta}{(r_b + r_c)}\right] \tag{A.43}$$

The TOF measurements and signal models of each transmitter and receiver pair are:

$$t_{aa} = \frac{2r_a}{c} \tag{A.44}$$

$$t_{ab} = t_{ba} = \frac{r_1 + r_2}{c} \tag{A.45}$$

$$t_{bb} = \frac{2r_b}{c} \tag{A.46}$$

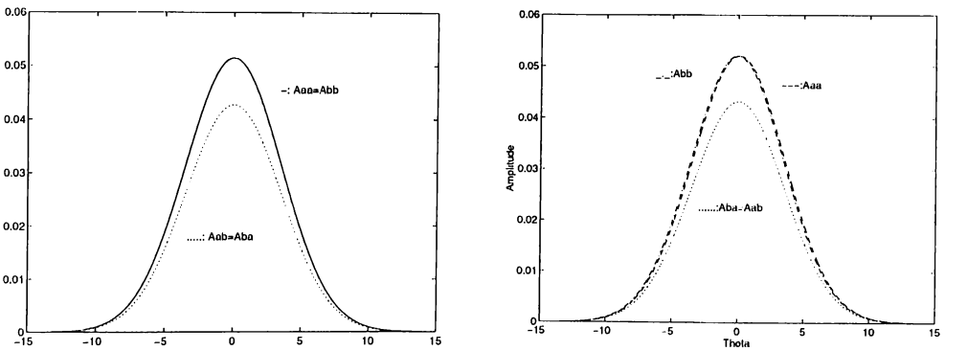
$$A_{aa}(r,\theta,d,t) = \rho_c \frac{A_{max} r_{min}^{3/2}}{r_a^{3/2}} e^{-\frac{\theta_a^2}{\sigma_T^2}} e^{-\frac{(t - t_{aa} - \frac{3}{f_o})^2}{\sigma^2}} \sin[2\pi f_0(t - t_{aa})] \tag{A.47}$$

$$A_{ab}(r,\theta,d,t) = \rho_c \frac{A_{max} r_{min}^{3/2}}{r_1 r_2^{1/2}} e^{-\frac{\theta_1^2 + \theta_2^2}{2\sigma_T^2}} e^{-\frac{(t - t_{ab} - \frac{3}{f_o})^2}{\sigma^2}} \sin[2\pi f_0(t - t_{ab})] \tag{A.48}$$

$$A_{ba}(r,\theta,d,t) = \rho_c \frac{A_{max} r_{min}^{3/2}}{r_2 r_1^{1/2}} e^{-\frac{\theta_2^2 + \theta_1^2}{2\sigma_T^2}} e^{-\frac{(t - t_{ba} - \frac{3}{f_o})^2}{\sigma^2}} \sin[2\pi f_0(t - t_{ba})] \tag{A.49}$$

$$A_{bb}(r,\theta,d,t) = \rho_c \frac{A_{max} r_{min}^{3/2}}{r_b^{3/2}} e^{-\frac{\theta_b^2}{\sigma_T^2}} e^{-\frac{(t - t_{bb} - \frac{3}{f_o})^2}{\sigma^2}} \sin[2\pi f_0(t - t_{bb})] \tag{A.50}$$

where ρ_c is the reflection coefficient of the cylinder which depends on its radius of curvature. In the limit as r_c goes to zero, a line target is obtained. On the other hand, as r_c goes to infinity, the cylinder becomes a planar target. According to the amplitude curves in Figure A.6, obtained by using the above models, when $r_c \geq 10$ m the amplitude curves of the transmitter and receiver pairs are the same as these curves obtained when the target is a plane.



(a) (b)

Figure A.6: Amplitude curves of the detected signal in the case of (a) planar target (b) cylindrical target with radius 10 m.

When radius of the cylinder increases, the amplitude of detected signal increases. An approximate model is to vary ρ_c linearly with the radius of the cylinder based on the given data points. A better model would result if more data points were taken and the best fitting curve was found. In the upper limit, if r_c is larger than 10 m, then the cylinder behaves like a planar target. As a result,

$$\lim_{r_c \to \infty} \rho_c \frac{A_{max} r_{min}^{3/2}}{r_a^{3/2}} = \frac{A_{max} r_{min}}{r_a} \tag{A.51}$$

In the lower limiting case, if $r_c \leq 5$ cm, the cylinder behaves as a line target which can be treated as an edge with $\theta_e \leq 30^o$. Then, $\rho_c \cong \rho_e$. By using Equation A.33, ρ_e is calculated as 1.98×10^{-6} for $\theta_e = 30^o$. According to the above argument, ρ_c can be expressed as:

$$\rho_c = 0.44946 r_c - 0.022471 \tag{A.52}$$

where r_c is in meters.

Moreover, it can be seen from the equations of detected signals for both the edge and the cylinder that they produce a cylindrically-divergent echo wavefront that decays as $r^{-1/2}$. In both cases, the received signal is only

composed of the diffracted signal. Since this diffracted component attenuates very quickly, reflections from edges and cylinders with small cross-sections are the most difficult echoes to detect.

A.5 Acute Corner Model

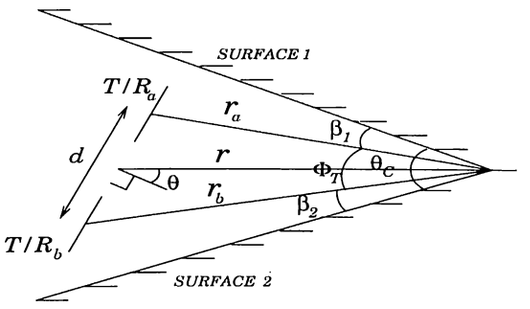


Figure A.7: Geometry of the problem when the target is an acute corner.

An *acute corner* is an intersection of two planes that form a concave space when observed from the transducers. Wedge angle θ_c of acute corner ranges from 0 to 90°. From the geometry of Figure A.7, β, ϕ_T, r_a and r_b can be expressed as:

$$\beta_1 + \beta_2 = \theta_c - \phi_T \tag{A.53}$$

$$\phi_T = \cos^{-1}\left(\frac{r_a^2 + r_b^2 - d^2}{2r_a r_b}\right) \tag{A.54}$$

$$r_a = \sqrt{r^2 + \frac{d^2}{4} - dr\sin\theta} \tag{A.55}$$

$$r_b = \sqrt{r^2 + \frac{d^2}{4} + dr\sin\theta} \tag{A.56}$$

As shown in Figure A.8, when $\theta = \theta_1 = 90° - \gamma$, transducer pair sees the second surface perpendicularly. Therefore, when $\theta \in [\theta_1 - \theta_o, \theta_1 + \theta_o]$, this transducer pair sees the second surface as a planar target. As a result, model of the acute corner when θ is in this range is the same as that of a plane with

$$r' = r\sin(\theta_c - \gamma) \quad \text{and} \quad \theta' = \theta_1 - \theta \tag{A.57}$$

Moreover, the same argument is true for $\theta \in [-(\theta_1 + \theta_o), -(\theta_1 - \theta_o)]$ with

$$r' = r\sin(\gamma) \quad \text{and} \quad \theta' = \theta_1 + \theta \tag{A.58}$$

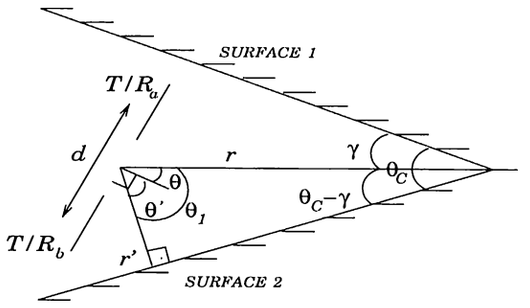


Figure A.8: Geometry of the problem when the target is an acute corner and $\theta \in [\theta_1 - \theta_o, \theta_1 + \theta_o]$.

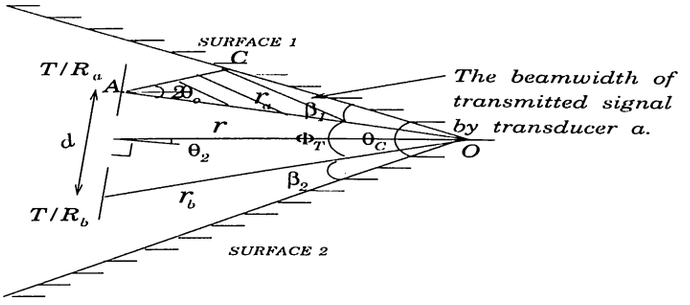


Figure A.9: Geometry of the problem when the target is an acute corner with critical angle θ_2.

When θ ranges from $\theta_1 - \theta_o$ to θ_2 in Figure A.9, the transmitted signal from each transducer first reaches the second surface and then the first surface. By applying sine and cosine theorems to the triangle ACO in Figure A.9, the following equation is obtained:

$$\frac{\sin^2\beta - \sin^2(2\theta_o) + \sin^2(\beta + 2\theta_o)}{\sin(\beta + 2\theta_o)} = 2\sin\beta\cos(2\theta_o) \qquad \text{(A.59)}$$

Since β is a function of ϕ_T which is a function of r_a and r_b with θ_2 dependence by Equations A.53 and A.54, it is not possible to solve Equation A.59 analytically which involves only the unknown θ_2. Numerical methods must be used to find the solution to this equation.

According to the above, r_{ab} which is the distance that the signal travels from transmitter a to receiver b, α_{ab} which is the angle from the signal transmitted by transmitter a and received by receiver b to the line-of-sight of transducer a, and β_{ab} which is the angle from this signal to the line-of-sight of transducer b, are found by the help of the geometry in Figure A.10. In the analysis, each surface of the acute corner is considered a mirror reflector.

Referring to Figure A.10, the distances and angles for each transmitter and

receiver pair are:

$$r_{aa} = 2r_a \sin\theta_c \tag{A.60}$$

$$\alpha_{aa} = 90^o - \theta_c + \theta_a \tag{A.61}$$

$$\beta_{aa} = 90^o - \theta_c - \theta_a \tag{A.62}$$

$$r_{ab} = \sqrt{r_a^2 + r_b^2 - 2r_a r_b \cos(2\theta_c - \phi_T)} \tag{A.63}$$

$$\alpha_{ab} = \cos^{-1}\left(\frac{r_{ab}^2 + d^2 - r_{bb}^2}{2dr_{ab}}\right) - 90^o \tag{A.64}$$

$$\beta_{ab} = \cos^{-1}\left(\frac{r_{ab}^2 + d^2 - r_{aa}^2}{2dr_{ab}}\right) - 90^o \tag{A.65}$$

$$r_{ba} = \sqrt{r_a^2 + r_b^2 - 2r_a r_b \cos(2\theta_c + \phi_T)} \tag{A.66}$$

$$\alpha_{ba} = 90^o - \cos^{-1}\left(\frac{r_{ba}^2 + d^2 - r_{aa}^2}{2dr_{ba}}\right) \tag{A.67}$$

$$\beta_{ba} = 90^o - \cos^{-1}\left(\frac{r_{ba}^2 + d^2 - r_{bb}^2}{2dr_{ba}}\right) \tag{A.68}$$

$$r_{bb} = 2r_b \sin\theta_c \tag{A.69}$$

$$\alpha_{bb} = 90^o - \theta_c + \theta_b \tag{A.70}$$

$$\beta_{bb} = 90^o - \theta_c - \theta_b \tag{A.71}$$

All distances are in meters and all angles are in degrees.

Note that in Equations A.60 and A.69 the round-trip distances measured by the sensors are twice the true range when the target is a right-angle corner but are less than twice the true range by a factor of $\sin\theta_c$ if the target is an acute corner with parameter θ_c. This result is significant since regardless of how the sensor is oriented with respect to the line defining the acute corner, as

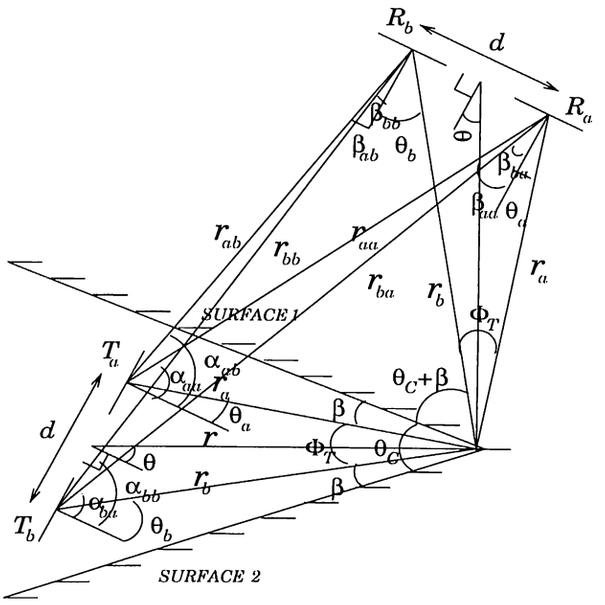


Figure A.10: Geometry of the problem when each surface of acute corner is considered a specular reflector when $\theta \in [\theta_1 - \theta_o, \theta_2)$.

long as it is at a distance r_a (or r_b), the same round-trip distance measurement will be obtained. Such points which give the same reading fall onto an arc bounded by the acute corner with center at the junction point.

In addition, since the problem is symmetric, same argument is true when $\theta \in (-\theta_2, -(\theta_1 - \theta_o)]$. In this case, the transmitted signal from each transducer first reaches surface 1 and then surface 2. For this case, the unknown ranges and angles can be found as:

$$r_{ab} = \sqrt{r_a^2 + r_b^2 - 2r_a r_b \cos(2\theta_c - \phi_T)} \tag{A.72}$$

$$\alpha_{ab} = 90^o - \cos^{-1}\left(\frac{r_{ab}^2 + d^2 - r_{bb}^2}{2dr_{ab}}\right) \tag{A.73}$$

$$\beta_{ab} = 90^o - \cos^{-1}\left(\frac{r_{ab}^2 + d^2 - r_{aa}^2}{2dr_{ab}}\right) \tag{A.74}$$

$$r_{ba} = \sqrt{r_a^2 + r_b^2 - 2r_a r_b \cos(2\theta_c - \phi_T)} \tag{A.75}$$

$$\alpha_{ba} = \cos^{-1}\left(\frac{r_{ba}^2 + d^2 - r_{aa}^2}{2dr_{ba}}\right) - 90^o \tag{A.76}$$

$$\beta_{ba} = \cos^{-1}\left(\frac{r_{ba}^2 + d^2 - r_{bb}^2}{2dr_{ba}}\right) - 90^o \tag{A.77}$$

Note that, $r_{aa}, \alpha_{aa}, \beta_{aa}, r_{bb}, \alpha_{bb}, \beta_{bb}$ are the same as in the previous case.

When θ is in the interval $[-\theta_2, \theta_2]$ the signal transmitted by transducer a initially reaches surface 1 and then surface 2, while the signal transmitted by transducer b first reaches surface 2 and then the other surface. From the geometry given in Figure A.11, r_{ij}'s, α_{ij}'s and β_{ij}'s are found as for $\theta \in [-\theta_2, 0)$:

$$r_{ab} = r_{ba} = \sqrt{r_a^2 + r_b^2 - 2r_a r_b \cos(2\theta_c - \phi_T)} \tag{A.78}$$

$$r_{aa} = 2r_a \sin\theta_c \tag{A.79}$$

$$r_{bb} = 2r_b \sin\theta_c \tag{A.80}$$

$$\alpha_{aa} = 90^o - \theta_c + \theta_a \tag{A.81}$$

$$\beta_{aa} = 90^o - \theta_c - \theta_a \tag{A.82}$$

$$\alpha_{bb} = 90^o - \theta_c - \theta_b \tag{A.83}$$

$$\beta_{bb} = 90^o - \theta_c + \theta_b \tag{A.84}$$

$$\alpha_{ab} = \cos^{-1}\left(\frac{r_{ab}^2 + d^2 - r_{bb}^2}{2dr_{ab}}\right) - 90^o \tag{A.85}$$

$$\beta_{ab} = \cos^{-1}\left(\frac{r_{ab}^2 + d^2 - r_{aa}^2}{2dr_{ab}}\right) - 90^o \tag{A.86}$$

$$\alpha_{ba} = \cos^{-1}\left(\frac{r_{ba}^2 + d^2 - r_{aa}^2}{2dr_{ba}}\right) - 90^o \tag{A.87}$$

$$\beta_{ba} = \cos^{-1}\left(\frac{r_{ba}^2 + d^2 - r_{bb}^2}{2dr_{ba}}\right) - 90^o \tag{A.88}$$

For $\theta \in (0, \theta_2]$:

$$r_{ab} = r_{ba} = \sqrt{r_a^2 + r_b^2 - 2r_a r_b \cos(2\theta_c - \phi_T)} \tag{A.89}$$

$$r_{aa} = 2r_a \sin\theta_c \tag{A.90}$$

$$r_{bb} = 2r_b \sin\theta_c \tag{A.91}$$

$$\alpha_{aa} = 90^o - \theta_c - \theta_a \tag{A.92}$$

$$\beta_{aa} = 90^o - \theta_c + \theta_a \tag{A.93}$$

$$\alpha_{bb} = 90^o - \theta_c + \theta_b \tag{A.94}$$

$$\beta_{bb} = 90^o - \theta_c - \theta_b \tag{A.95}$$

$$\alpha_{ab} = \cos^{-1}\left(\frac{r_{ab}^2 + d^2 - r_{bb}^2}{2dr_{ab}}\right) - 90^o \tag{A.96}$$

$$\beta_{ab} = \cos^{-1}\left(\frac{r_{ab}^2 + d^2 - r_{aa}^2}{2dr_{ab}}\right) - 90^o \tag{A.97}$$

$$\alpha_{ba} = \cos^{-1}\left(\frac{r_{ba}^2 + d^2 - r_{aa}^2}{2dr_{ba}}\right) - 90^o \tag{A.98}$$

$$\beta_{ba} = \cos^{-1}\left(\frac{r_{ba}^2 + d^2 - r_{bb}^2}{2dr_{ba}}\right) - 90^o \tag{A.99}$$

When $\theta = 0^o$:

$$r_{ab} = r_{ba} = 2r_a \sin\left(\theta_c - \frac{\phi_T}{2}\right) \tag{A.100}$$

$$r_{aa} = 2r_a \sin\theta_c \tag{A.101}$$

$$r_{bb} = 2r_b \sin\theta_c \tag{A.102}$$

$$\alpha_{aa} = \alpha_{bb} = 90^o - \theta_c + \theta_a \tag{A.103}$$

$$\beta_{aa} = \beta_{bb} = 90^o - \theta_c - \theta_a \tag{A.104}$$

$$\alpha_{ab} = \alpha_{ba} = \beta_{ab} = \beta_{ba} = 90^o - \theta_c - \theta_a + \frac{\phi_T}{2} \tag{A.105}$$

Note that $r_a = r_b$ and $\theta_a = \theta_b$ in this case.

According to the distances and inclination angles as found above, the TOF measurements and detected signals for each transmitter and receiver pair when the target is an acute corner can be written as:

$$t_{aa} = \frac{r_{aa}}{c} \tag{A.106}$$

$$t_{ab} = \frac{r_{ab}}{c} \tag{A.107}$$

$$t_{ba} = \frac{r_{ba}}{c} \tag{A.108}$$

$$t_{bb} = \frac{r_{bb}}{c} \tag{A.109}$$

$$A_{aa}(r,\theta,d,t) = 2\frac{A_{max}r_{min}}{r_{aa}} e^{-\frac{\alpha_{aa}^2+\beta_{aa}^2}{2\sigma_T^2}} e^{-\frac{(t-t_{aa}-\frac{3}{f_o})^2}{\sigma^2}} \sin[2\pi f_o(t-t_{aa})] \tag{A.110}$$

$$A_{ab}(r,\theta,d,t) = 2\frac{A_{max}r_{min}}{r_{ab}} e^{-\frac{\alpha_{ab}^2+\beta_{ab}^2}{2\sigma_T^2}} e^{-\frac{(t-t_{ab}-\frac{3}{f_o})^2}{\sigma^2}} \sin[2\pi f_o(t-t_{ab})] \tag{A.111}$$

$$A_{ba}(r,\theta,d,t) = 2\frac{A_{max}r_{min}}{r_{ba}} e^{-\frac{\alpha_{ba}^2+\beta_{ba}^2}{2\sigma_T^2}} e^{-\frac{(t-t_{ba}-\frac{3}{f_o})^2}{\sigma^2}} \sin[2\pi f_o(t-t_{ba})] \tag{A.112}$$

$$A_{bb}(r,\theta,d,t) = 2\frac{A_{max}r_{min}}{r_{bb}} e^{-\frac{\alpha_{bb}^2+\beta_{bb}^2}{2\sigma_T^2}} e^{-\frac{(t-t_{bb}-\frac{3}{f_o})^2}{\sigma^2}} \sin[2\pi f_o(t-t_{bb})] \tag{A.113}$$

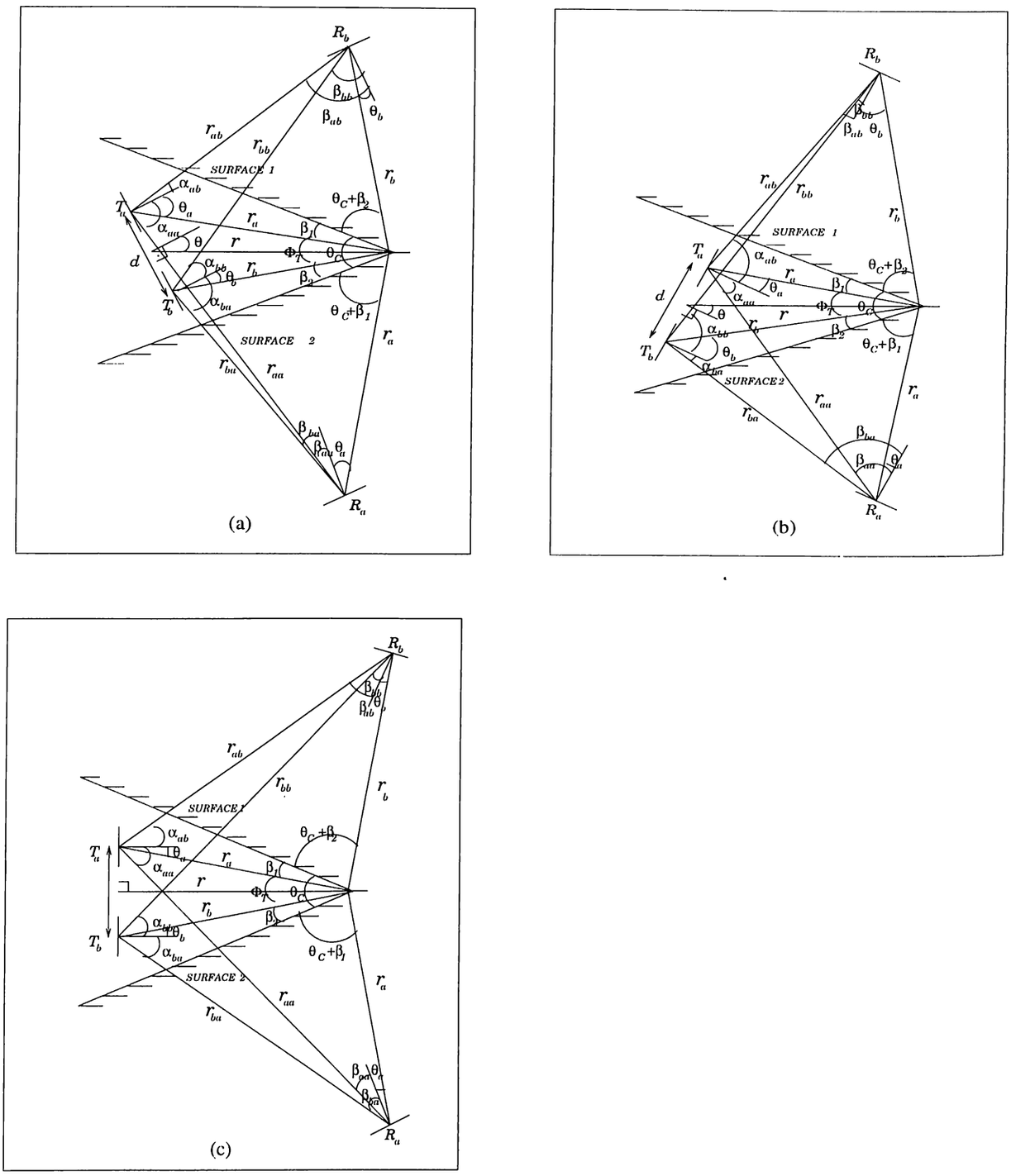


Figure A.11: Geometry of the problem when each surface of acute corner is considered a specular reflector and $\theta \in [-\theta_2, \theta_2]$. (a) $\theta \in [-\theta_2, 0)$, (b) $\theta \in (0, \theta_2]$, (c) $\theta = 0^o$.

A.6 Effect of Transducer Separation and Range on TOF and Amplitude Characteristics

The ratio of transducer separation to operating range (d/r) is an important parameter in the differentiation of target primitives directly affecting how well these target primitives can be differentiated by their TOF and amplitude characteristics. The further apart are the transducers, the larger are the differentials in TOF and amplitude as long as the sensitivity patterns of the targets overlap. This supports improved differentiation of the targets. In the limit as $\frac{d}{r} \rightarrow 0$ which corresponds to the case that either the targets are too close together or the target is too far, the transducers behave as a single transducer and the differential signals are not reliable. This situation is equivalent to the case of trying to differentiate the targets using a single transducer at a fixed location, which is not feasible as shown in [13].

Differentials in TOF and amplitude versus sensor separation characteristics at a fixed inclination angle θ for all target primitives used in this study are given in Figures A.13–A.14 for three different range values. The fixed inclination angle θ for each target at which maximum amplitude differentials are obtained when $r = 0.5$ m and $d = 5$ cm are given in Table A.1.

Target Type	θ
plane	0^o
corner	4.2^o
edge of $\theta_e = 90^o$	3.6^o
cylinder of $r_c = 15$ cm	3.0^o
acute corner of $\theta_c = 60^o$	0.5^o
acute corner of $\theta_c = 45^o$	0.5^o
acute corner of $\theta_c = 30^o$	1.0^o

Table A.1: The fixed inclination angle θ for all target primitives.

In the limit as range goes to infinity, transducer pair can be treated as a single transducer, therefore a reduction in TOF differentials is expected. Referring to Figures A.13–A.14, for a given transducer separation, the differentials in TOF estimates are inversely proportional to range for all target primitives as expected except for the edge type target primitive. This controversial result

for an edge can be proven by investigating the behavior of the differential TOF. The expression for differential TOF is given by:

$$\triangle t = t_{aa} - t_{ab} = \frac{1}{c}\left(\sqrt{r^2 + \frac{d^2}{4} - dr\sin\theta} - \sqrt{r^2 + \frac{d^2}{4} + dr\sin\theta}\right) \quad \text{(A.114)}$$

Evaluating the derivative of the above equation with respect to r;

$$\frac{\partial \triangle t}{\partial r} = \frac{r - \frac{d}{2}\sin\theta}{\sqrt{r^2 + \frac{d^2}{4} - dr\sin\theta}} - \frac{r + \frac{d}{2}\sin\theta}{\sqrt{r^2 + \frac{d^2}{4} + dr\sin\theta}} \quad \text{(A.115)}$$

which can be rewritten as

$$\frac{\partial \triangle t}{\partial r} = \frac{r - \frac{d}{2}\sin\theta}{\sqrt{\left(r - \frac{d}{2}\sin\theta\right)^2 + \frac{d^2}{4}\cos^2\theta}} - \frac{r + \frac{d}{2}\sin\theta}{\sqrt{\left(r + \frac{d}{2}\sin\theta\right)^2 + \frac{d^2}{4}\cos^2\theta}} \quad \text{(A.116)}$$

or

$$\frac{\partial \triangle t}{\partial r} = \frac{x}{\sqrt{x^2 + \triangle}} - \frac{y}{\sqrt{y^2 + \triangle}} \quad \text{(A.117)}$$

where $x = r - \frac{d}{2}\sin\theta$ and $y = r + \frac{d}{2}\sin\theta$.

Now, if $\theta \geq 0^o$, then $x \leq y$ and $\frac{\partial \triangle t}{\partial r} \leq 0$. If $\theta < 0^o$, then $x > y$ and $\frac{\partial \triangle t}{\partial r} > 0$. In the first case, any positive change $\triangle r$ in r will result in a negative change in $\triangle t$ which is in accordance with the result in Figure A.13(e) where $\theta = 3.6^o$. In the second case, a positive change in r will result in a positive change in the TOF.

The limit of the TOF differential as r goes to zero and infinity can be found as:

$$\lim_{r\to 0} \triangle t = \lim_{r\to 0} \frac{1}{c}\left(\sqrt{r^2 + \frac{d^2}{4} - dr\sin\theta} - \sqrt{r^2 + \frac{d^2}{4} + dr\sin\theta}\right) \quad \text{(A.118)}$$

$$\lim_{r\to 0} \triangle t = 0 \quad \text{(A.119)}$$

and

$$\lim_{r\to\infty} \triangle t = \lim_{r\to\infty} \frac{r}{c}\left(\sqrt{1 + \frac{d^2}{4r^2} - \frac{d}{r}\sin\theta} - \sqrt{1 + \frac{d^2}{4r^2} + \frac{d}{r}\sin\theta}\right) \quad \text{(A.120)}$$

For large values of r, $\frac{d}{r} << 1$ and the following approximation can be made:

$$\frac{ct_{aa}}{r} = \sqrt{1 + \frac{d^2}{4r^2} - \frac{d}{r}\sin\theta} \cong 1 + \frac{1}{2}\left(\frac{d^2}{4r^2} - \frac{d}{r}\sin\theta\right) \quad \text{(A.121)}$$

$$\frac{ct_{ab}}{r} = \sqrt{1 + \frac{d^2}{4r^2} + \frac{d}{r}\sin\theta} \cong 1 + \frac{1}{2}\left(\frac{d^2}{4r^2} + \frac{d}{r}\sin\theta\right) \quad \text{(A.122)}$$

and

$$\lim_{r\to\infty} \triangle t = -\frac{d}{c}\sin\theta \tag{A.123}$$

The results indicate that as r goes from zero to infinity, the differential in TOF estimates is nonzero for $\theta \neq 0^o$ and increases with r and $|\theta|$ (i.e. t_{ab} becomes greater than t_{aa} when $\theta > 0^o$ and t_{aa} becomes greater than t_{ab} when $\theta < 0^o$).

For all target primitives, at a fixed operating range, the differentials in TOF estimates increase with increasing sensor separation as expected (Figures A.13–A.14).

Echo amplitudes of all target primitives are inversely proportional to range. For the planar target, at fixed separation, differentials in echo amplitudes increase up to some critical d/r value which is around 0.3, then stay constant at some differential value which is equal to A_{aa} since $A_{ab} = 0$. This is because the sensitivity patterns of the two transducers which form the pair do not overlap if d is too large. This corresponds to the situation where the target falls within the sensitivity pattern of only one of the transducers or the sensitivity pattern of each transducer separately if the extent of the plane is long enough (Figure A.12(b)). Therefore, the amplitude of the signals $A_{ab}(r,\theta,d,t)$ and $A_{ba}(r,\theta,d,t)$ are zero. For the corner target, amplitude differential increases up to some critical d/r value which is around 0.15, then it begins to decrease. After a critical d/r value which is around 0.45, it stays constant at a negative differential value which is equal to $-A_{ab}$ since A_{aa} is zero. This corresponds the inverse situation of the planar target case due to the position of the virtual receivers discussed before.

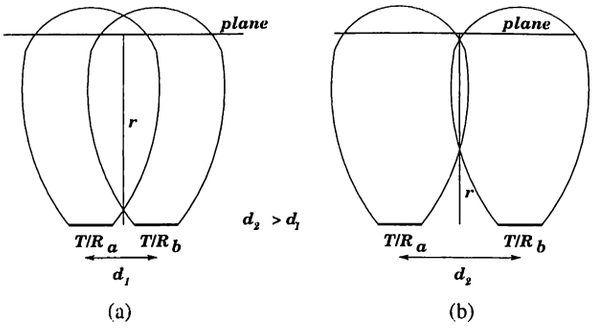


Figure A.12: A planar target falls (a) within the intersection of the sensitivity patterns of both transducers (b) outside the intersection of the sensitivity patterns so that cross signals are not detected.

For the edge and cylindrical target primitives, differentials in amplitude for a given fixed separation again increase up to some critical d/r value z_1, then

they begin to decrease after a critical d/r value z_2 and stay constant at zero. In this case, the targets fall within neither the sensitivity patterns of a single transducer nor the intersection of sensitivity patterns of the two transducers which form the pair. These critical values z_1 and z_2 for edge and cylinder are given in Table A.2.

Target Type	z_1	z_2
edge of $\theta_e = 90^o$	0.175	0.5
cylinder of $r_c = 15$ cm	0.125	0.55

Table A.2: The critical d/r values for edge and cylindrical target primitives.

For acute corners, the amplitude differentials are plotted up to the maximum separation which can be reached at the fixed operating range due to the physical limitation by the two planes forming the acute corner on each side of the sensor. This maximum separation corresponds to $2\left(r\tan\frac{\theta_c}{2} - a\right)$. Up to this separation, the characteristics of amplitude differentials increase until d/r is around 0.2, then they begin to decrease.

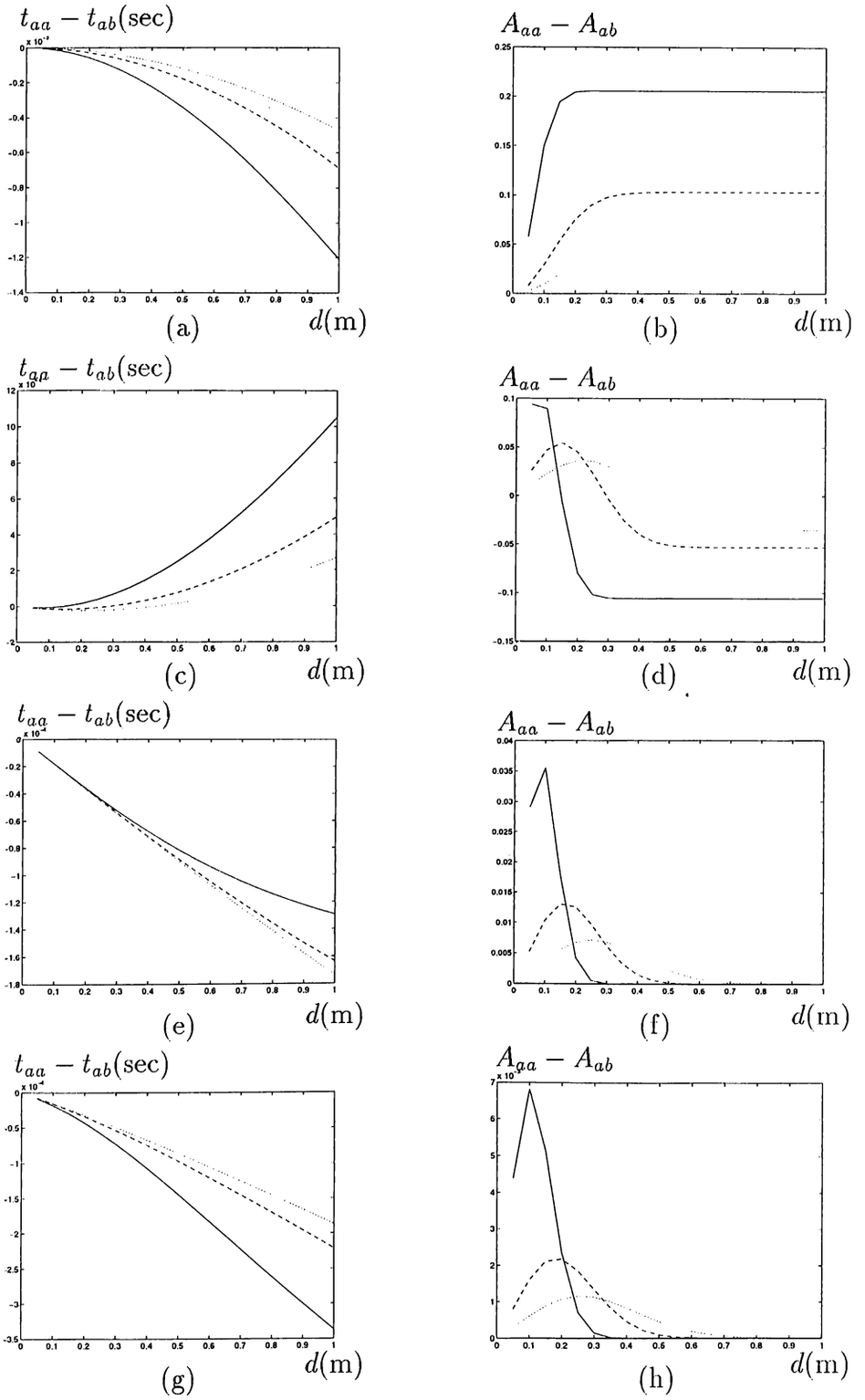


Figure A.13: TOF and amplitude differentials versus transducer separation at various range values for (a)-(b) plane (c)-(d) corner (e)-(f) edge of $\theta_e = 90^o$ (g)-(h) cylinder of $r_c = 15$ cm.

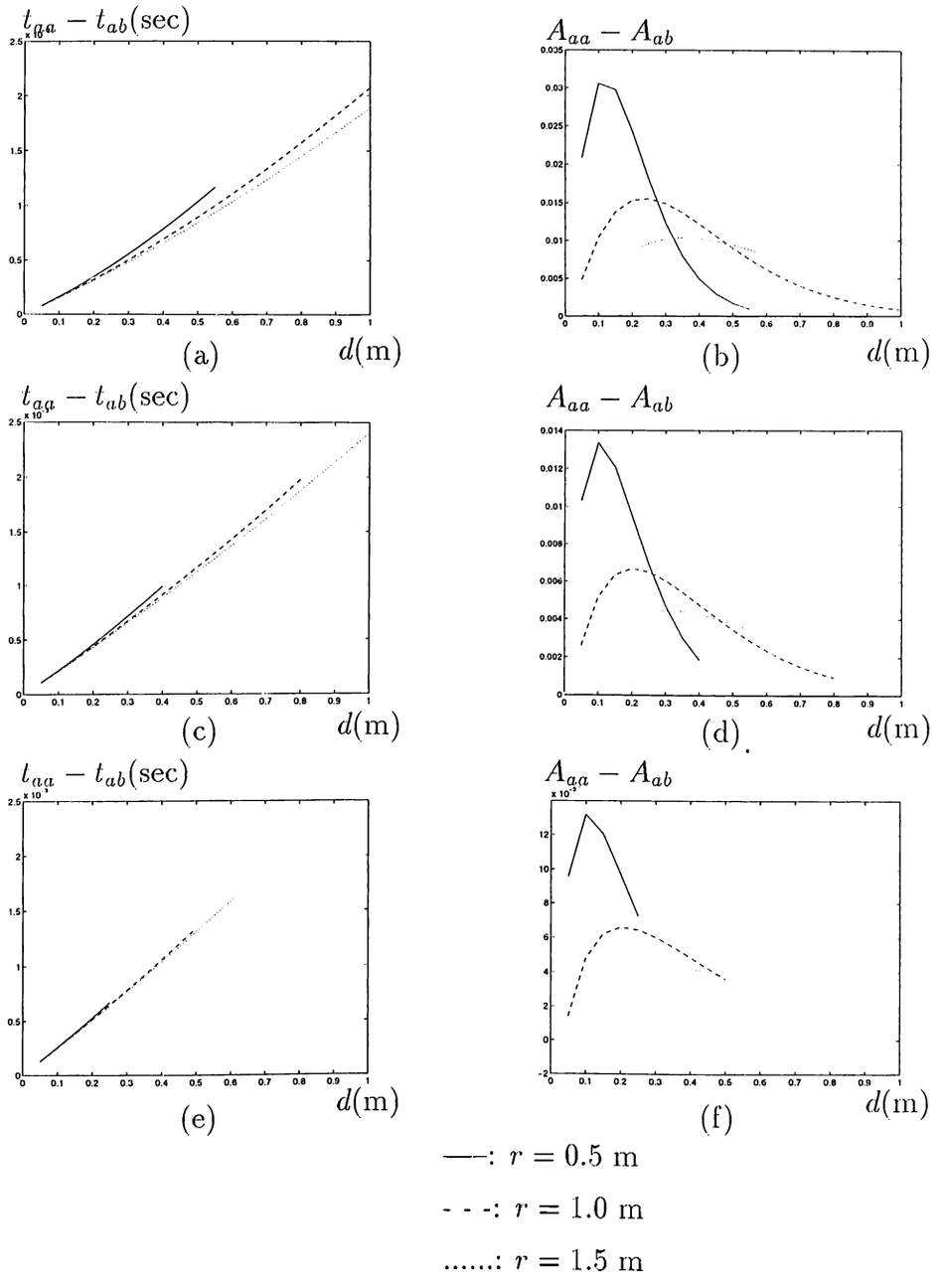


Figure A.14: Maximum TOF and amplitude differentials versus transducer separation at various range values for acute corners of (a)-(b) $\theta_c = 60^o$ (c)-(d) $\theta_c = 45^o$ (e)-(f) $\theta_c = 30^o$.

In this appendix, we have provided geometric reflection models for the target primitives. Based on these, we have given signal models for each target type which are supported by experimental data in Chapter 5. The effect of transducer separation on the signal characteristics has been investigated.

Appendix B

CHOICE OF NOISE MULTIPLICITY FACTOR k IN THE CLASSIFICATION ALGORITHM

In this appendix, echo amplitudes with different noise levels are presented. In order to make the differentiation algorithms robust to noise, we establish an appropriate value for the noise multiplicity factor k used in the algorithms via simulations.

Different levels of white Gaussian noise are added to the echo signal. As the results illustrate, amplitudes of detected signals for plane at range $r = 2$ m are indistinguishable from noise for $\sigma_A \geq 0.05$ which corresponds to 95% of maximum amplitudes of detected signals. The noisy amplitude curves are given in Figure B.1 for different noise standard deviations.

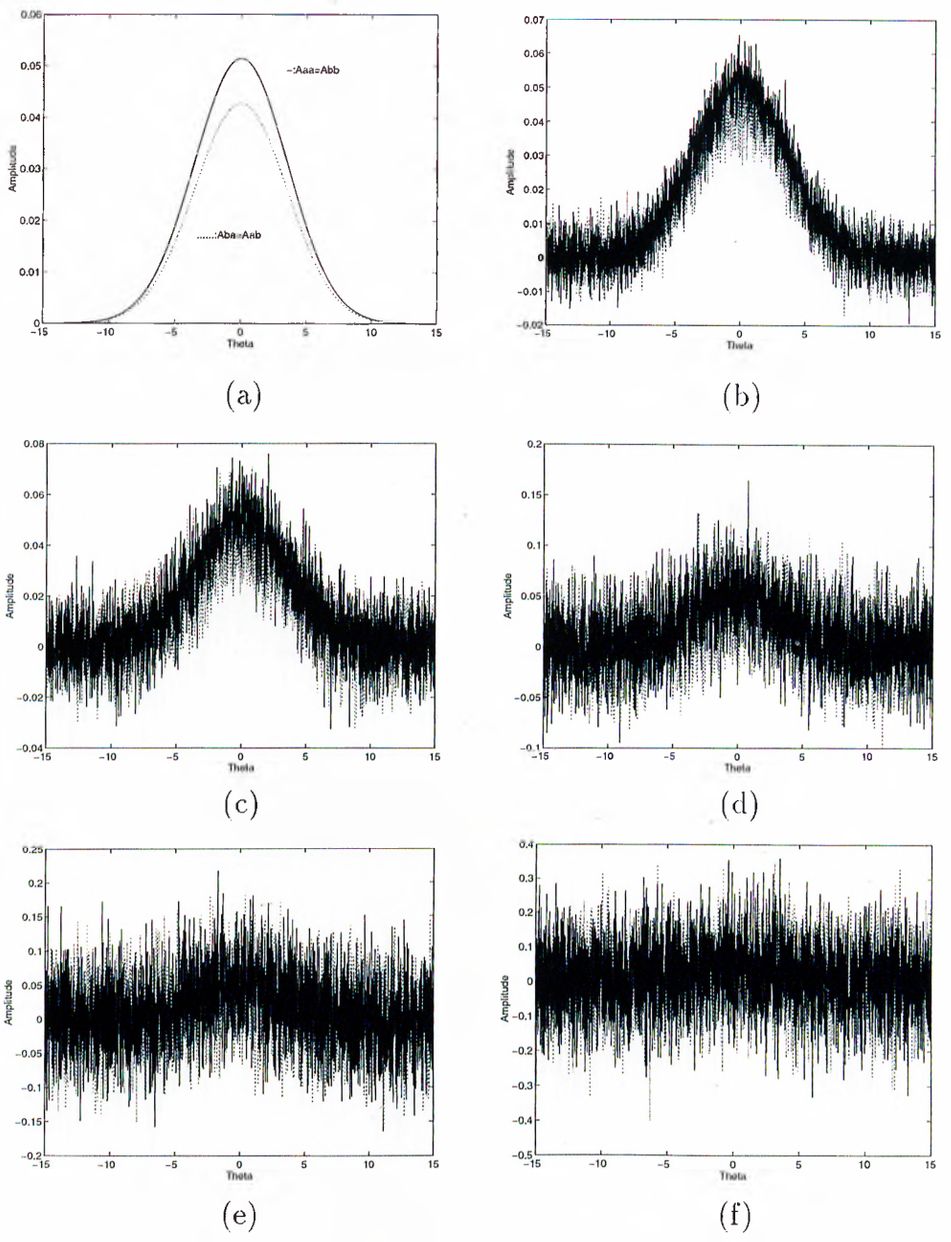


Figure B.1: Noisy amplitude for a plane at $r = 2$ m with noise standard deviation (a) $\sigma_A = 0$ (b) $\sigma_A = 0.005$ (c) $\sigma_A = 0.01$ (d) $\sigma_A = 0.03$ (e) $\sigma_A = 0.05$ (f) $\sigma_A = 0.1$.

However, maximum differences between the detected signal amplitudes which are used in the classification algorithm is around 0.008. Therefore, for noise standard deviation of 0.005 or larger, which corresponds to 62.5% of maximum amplitude difference in these signals, the differences in the signal levels are comparable to the noise and it becomes impossible to detect these differences. Noisy difference signals are given in Figure B.2 for various noise standard deviation values.

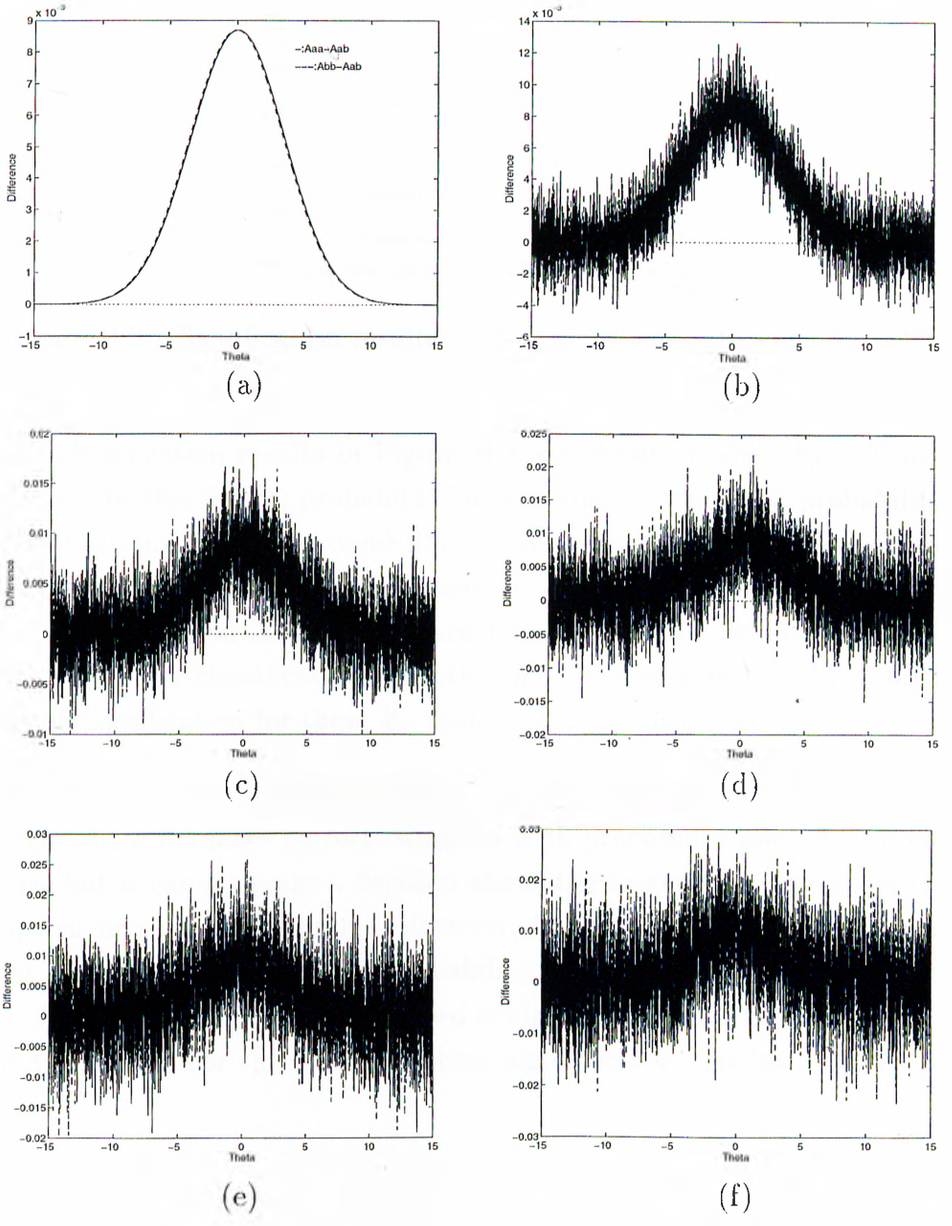


Figure B.2: Noisy amplitude for a plane at $r = 2$ m with noise standard deviation (a) $\sigma_A = 0$ (b) $\sigma_A = 0.001$ (c) $\sigma_A = 0.002$ (d) $\sigma_A = 0.003$ (e) $\sigma_A = 0.004$ (f) $\sigma_A = 0.005$.

For the reason given above, the noise standard deviations up to 0.008 are of interest in this study. For larger noise standard deviations, the performance of classification when excluding $k_A\sigma_A$ term in the algorithms (i.e. $k_A = 0$) is the same as that of a randomized decision rule where 50% of the time unknown target is guessed to be a plane, 50% of the time a corner [33]. This phenomenon can be seen in Figure B.3.

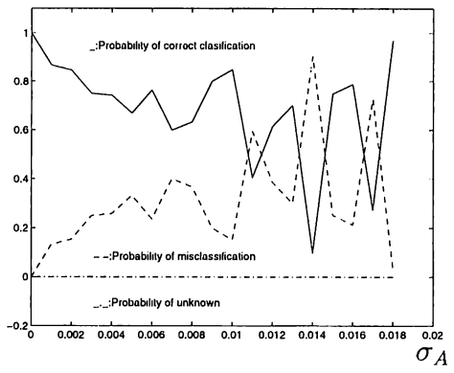


Figure B.3: Classification results of one transducer pair when $k_A = 0$.

The classification results in Figure B.3 are obtained when $k_A = 0$ in the algorithm. In this figure, probability of true classification and probability of misclassification oscillates around 0.5 for larger standard deviations of noise. Our problem is to choose the best value of k_A for the algorithm so that the result of classification is better for larger noise standard deviations. To find this k_A value, the classification algorithm has been run for $0 < k_A \leq 6$. The results of classification for these k_A values are given in Figure B.4.

Referring to the same figure for $k_A \geq 2$, classification algorithm works as a switch where it makes correct decisions with probability one when there is no noise but it cannot make a decision about the target when there is a small amount of noise in the system. However, for $k_A = 1$ the algorithm makes correct decisions with decreasing probability up to noise standard deviation of 0.005. After this value of noise standard deviation, it cannot make any decision about the target. For this reason in these algorithms k_A has been chosen as 1.

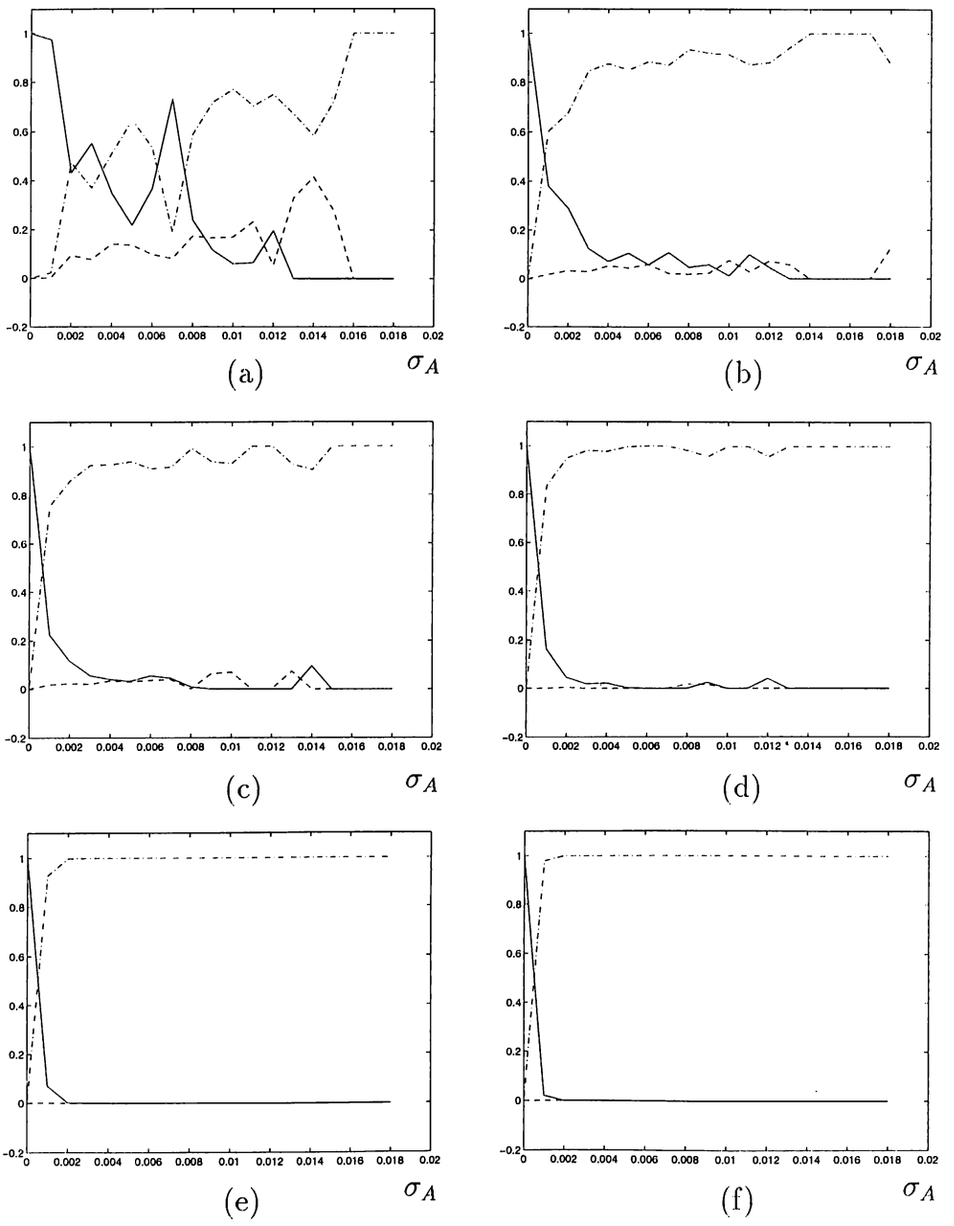


Figure B.4: Classification results for multiples of noise standard deviation, σ_A, with (a) $k_A = 0.5$ (b) $k_A = 1.0$ (c) $k_A = 1.5$ (d) $k_A = 2.0$ (e) $k_A = 3.0$ (f) $k_A = 4.0$.

Appendix C

RANGE ESTIMATION FOR AN ACUTE CORNER

The coefficients of the polynomial used to estimate range r for an acute corner at $\theta \neq 0^o$:

$$Ax^2 + Bx + C = 0 \quad \text{where} \quad x = 2r^2 + \frac{d^2}{2} \tag{C.1}$$

$$A = \left(\frac{r_{aa}^2 - r_{bb}^2}{r_{bb}^2}\right)^2 \tag{C.2}$$

$$B = \left(\frac{r_{aa}^2 + r_{bb}^2}{r_{bb}^2}\right)\left\{r_{aa}^2 - \frac{1}{r_{bb}^2}\left[(r_{aa}^2 + r_{bb}^2)(r_{ab}^2 + d^2) - (r_{ab}^2 - d^2)^2\right]\right\} \tag{C.3}$$

$$C = d^2 r_{ab}^2 \left(\frac{r_{aa}^2 + r_{bb}^2}{r_{bb}^2}\right)^2 \tag{C.4}$$

For $\theta = 0^o$:

$$r_a^2 = r_b^2 = r^2 + \frac{d^2}{4} \tag{C.5}$$

$$r_{aa} = r_{bb} \tag{C.6}$$

$$r_{ab} = 2r_b \sin\left(\theta_c - \frac{\phi_T}{2}\right) \tag{C.7}$$

$$\sin\theta_c = \frac{r_{bb}}{2r_b} \tag{C.8}$$

$$\cos\frac{\phi_T}{2} = \frac{r}{r_b} \tag{C.9}$$

$$\sin\frac{\phi_T}{2} = \frac{d}{2r_b} \tag{C.10}$$

To solve for r, an equation given below can be found with the help of above equation:

$$4(r_{ab}^2 + r_{bb}^2 - d^2)r^2 - 8r_{bb}r_{ab}r^2\sqrt{1 + \frac{d^2}{4r^2}} + d^2(r_{ab}^2 + r_{bb}^2 - d^2) = 0 \quad \text{(C.11)}$$

Since $r \gg d \Rightarrow \frac{d^2}{4r^2} \ll 1$, the approximation given below can be used in above equation:

$$\sqrt{1 + \frac{d^2}{4r^2}} \approx 1 + \frac{d^2}{8r^2} \quad \text{(C.12)}$$

Then,

$$r = \sqrt{\frac{d^2(r_{ab}r_{bb} + d^2 - r_{ab}^2 - r_{bb}^2)}{4(r_{ab} - r_{bb})^2 - d^2}} \quad \text{(C.13)}$$

Appendix D

AMPLITUDE CHARACTERISTICS OF EDGES AND CYLINDERS

In this appendix, amplitude characteristics of an edge with various values of angle θ_e, and that of a cylinder with various values of radius r_c are given in Figures D.1 and D.2 respectively.

Note that for the edge, as illustrated in Figure D.1, the relative amplitude characteristics is the same for different θ_e values. However, as θ_e increases from 30^o to 90^o, an increase in echo amplitude is observed. This is due to the increasing cross-section of the edge.

For the cylinder, as illustrated in Figure D.2, when the radius is relatively small, the characteristics is similar to that of an edge. As the radius increases, the characteristics resembles that of a plane quite well, especially for $r_c > 1$ m.

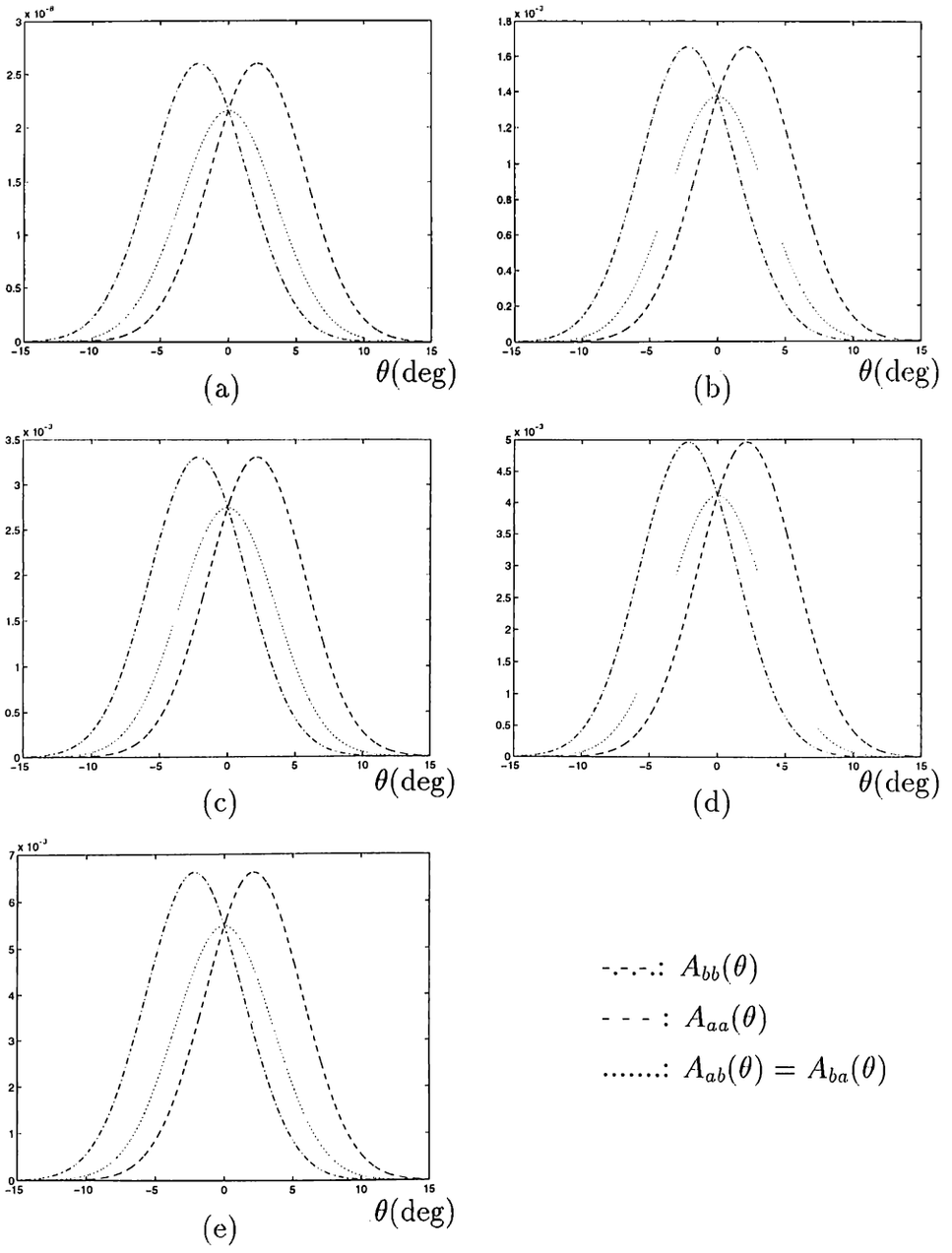


Figure D.1: Amplitude characteristics of an *edge* with (a) $\theta_e = 30^o$ (b) $\theta_e = 45^o$ (c) $\theta_e = 60^o$ (d) $\theta_e = 75^o$ (e) $\theta_e = 90^o$.

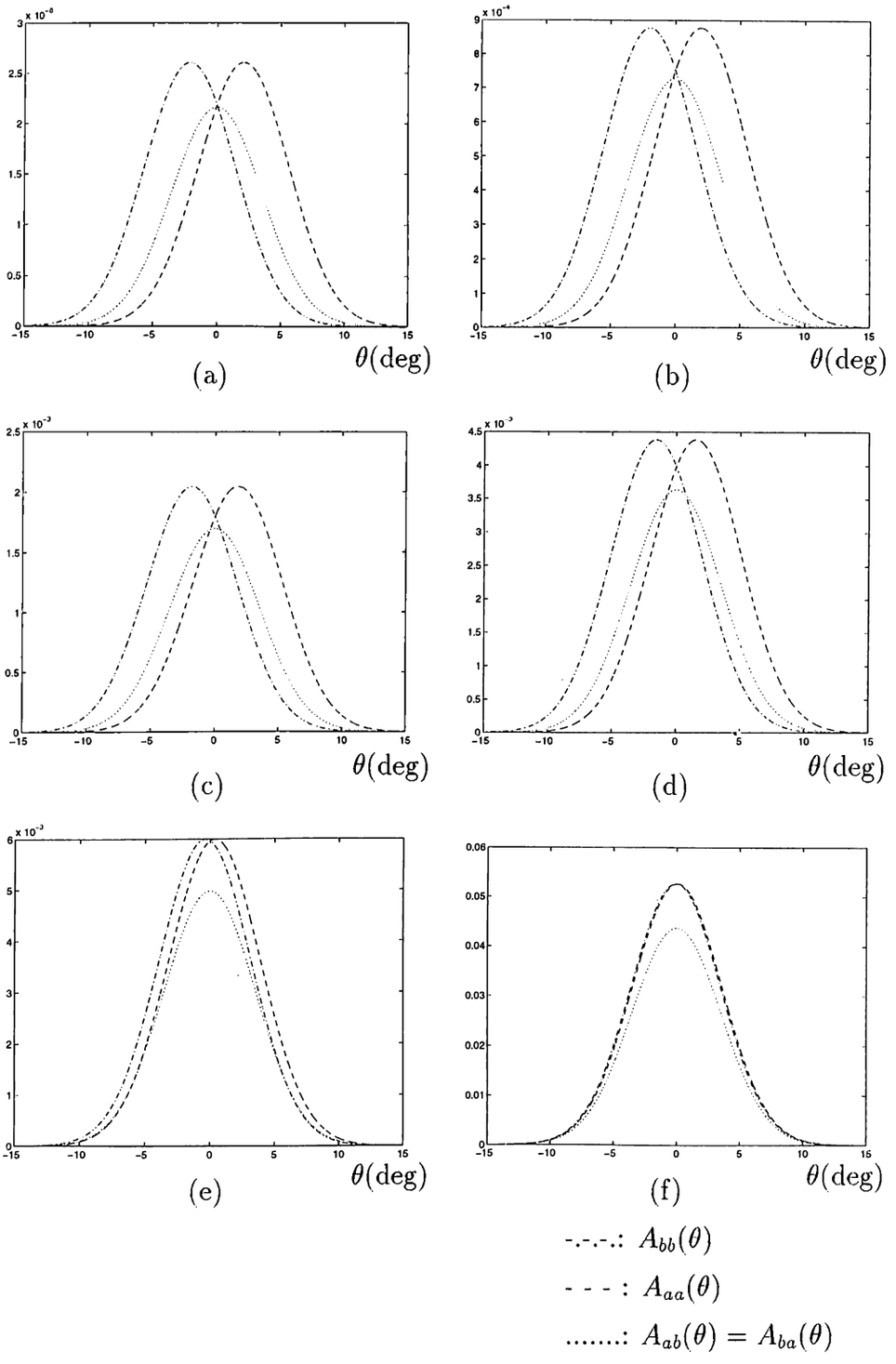


Figure D.2: Amplitude characteristics of a *cylinder* with (a) r_c = 5 cm (b) r_c = 20 cm (c) r_c = 40 cm (d) r_c = 80 cm (e) r_c = 1 m (f) r_c = 10 m.

Appendix E

RELATIONSHIP BETWEEN σ_A AND σ_t

In this appendix, the relationship between σ_A and σ_t is investigated. In the first part, the effect of change in threshold level and amplitude noise standard deviation on σ_t is studied with a typical sonar echo waveform. In the second part, a linear signal model of form $y(t) = at$ is used with different threshold settings and slopes under various amplitude noise standard deviations.

E.1 Relation between σ_A and σ_t in a Typical Sonar Echo

In this section, the effect of change in threshold level τ and amplitude noise standard deviation (σ_A) on the standard deviation in TOF estimate (σ_t) is investigated. A typical echo waveform for sonar produced by a plane is shown in Figure E.1.

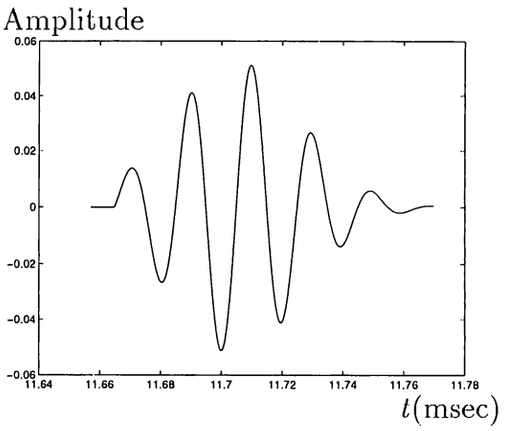


Figure E.1: A typical echo waveform.

As a first attempt, threshold level is kept constant at a level of 0.005 which is 9.5% of maximum echo amplitude produced by a planar target at $r = 2$ m. In the second step, threshold level is set to 5 times the amplitude noise standard deviation (i.e. $\tau = 5\sigma_A$). In both cases, the effect of varying σ_A in the intervals $[0, 0.02]$ and $[0, 0.008]$ respectively is observed over 1000 realizations on σ_t and m_t where m_t is the mean of TOF estimate. As a last step, threshold level is set as integer multiples of a constant amplitude noise standard deviation of 0.005 (i.e. $\tau = n\sigma_A$ where $\sigma_A = 0.005$). The effect of increasing the threshold level when σ_A is kept constant is observed on σ_t and m_t again over 1000 realizations. Standard deviation, mean and the bias of TOF estimate and biases in TOF measurements are given in Figures E.2 and E.3 respectively.

Referring to these figures, σ_t does not exceed 16 μsec for all amplitude noise standard deviations and threshold levels. Although TOF's are of the order of milliseconds, maximum bias in TOF estimates is around 40 μsec. The bias in TOF estimates increases dramatically with increasing σ_A when the threshold level is kept constant. However, when the threshold level is increased with the integer multiple of σ_A, the bias in TOF estimates decreases with increasing n and it is in the interval $[-1,1]$ μsec for $n \geq 5$.

In all of the simulation studies, threshold level is taken as 5 times the amplitude noise standard deviation. Moreover, amplitude noise standard deviations of 0.006 or larger correspond to 75% of maximum signal amplitude differences. In this case, differential signal levels are comparable to the noise and it becomes impossible to detect these differentials. In Figure 44, maximum value of σ_t is about 8 μsec and maximum bias in TOF estimates is around 12 μsec for $\sigma_A \in [0, 0.008]$ and a threshold level of $\tau = 5\sigma_A$. Since TOF's are around 11.6 msec, these biases in TOF estimates and standard deviation of TOF estimates are very small. In Table 3.4 in Chapter 3, maximum differences between TOF's

are of order of the 4×10^{-4} sec. Therefore, even if there exists an amplitude noise with standard deviation 0.008, corresponding to 100% of maximum signal amplitude difference, standard deviation in the TOF estimate is about 8 μsec which corresponds to 2% of maximum differences in the TOF's.

Some part of σ_t is contributed by the sampling period T_s [41]. Since TOF is a multiple of the sampling period, it has a random delay and this random delay is uniformly distributed in the interval $[0, T_s]$ where T_s is the sampling period. Standard deviation of this random delay in TOF estimates is $\sqrt{\frac{T_s^2}{12}}$, contributing to the standard deviation of TOF estimates. Sampling time, T_s, is taken as 1 μsec in all our simulation studies, therefore this contribution is 0.29 μsec.

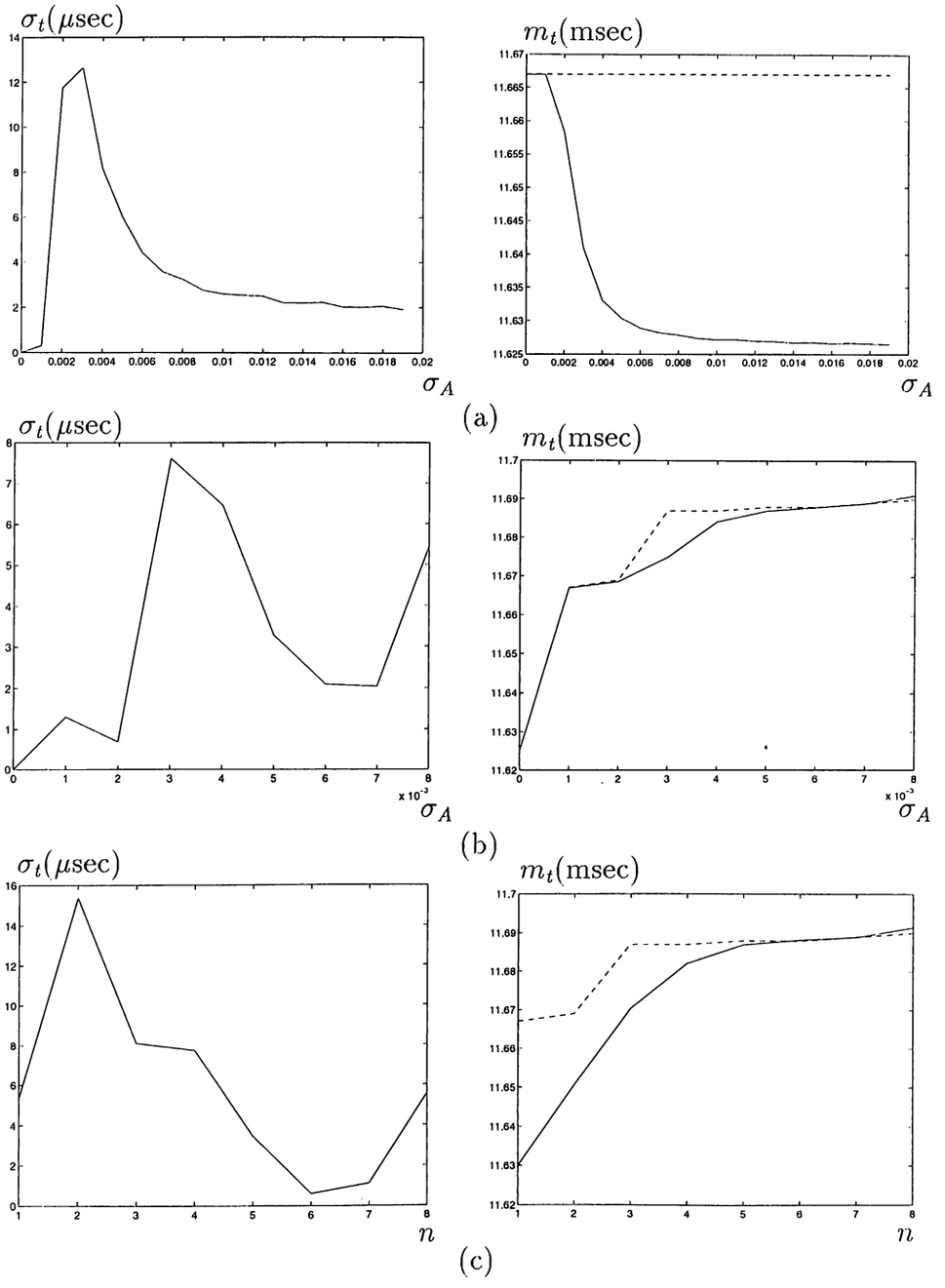


Figure E.2: Standard deviation and mean of TOF estimate over 1000 realizations when a typical echo waveform is used with (a) $\tau = 0.005$ (b) $\tau = 5\sigma_A$ (c) $\tau = n\sigma_A$ and $\sigma_A = 0.005$.

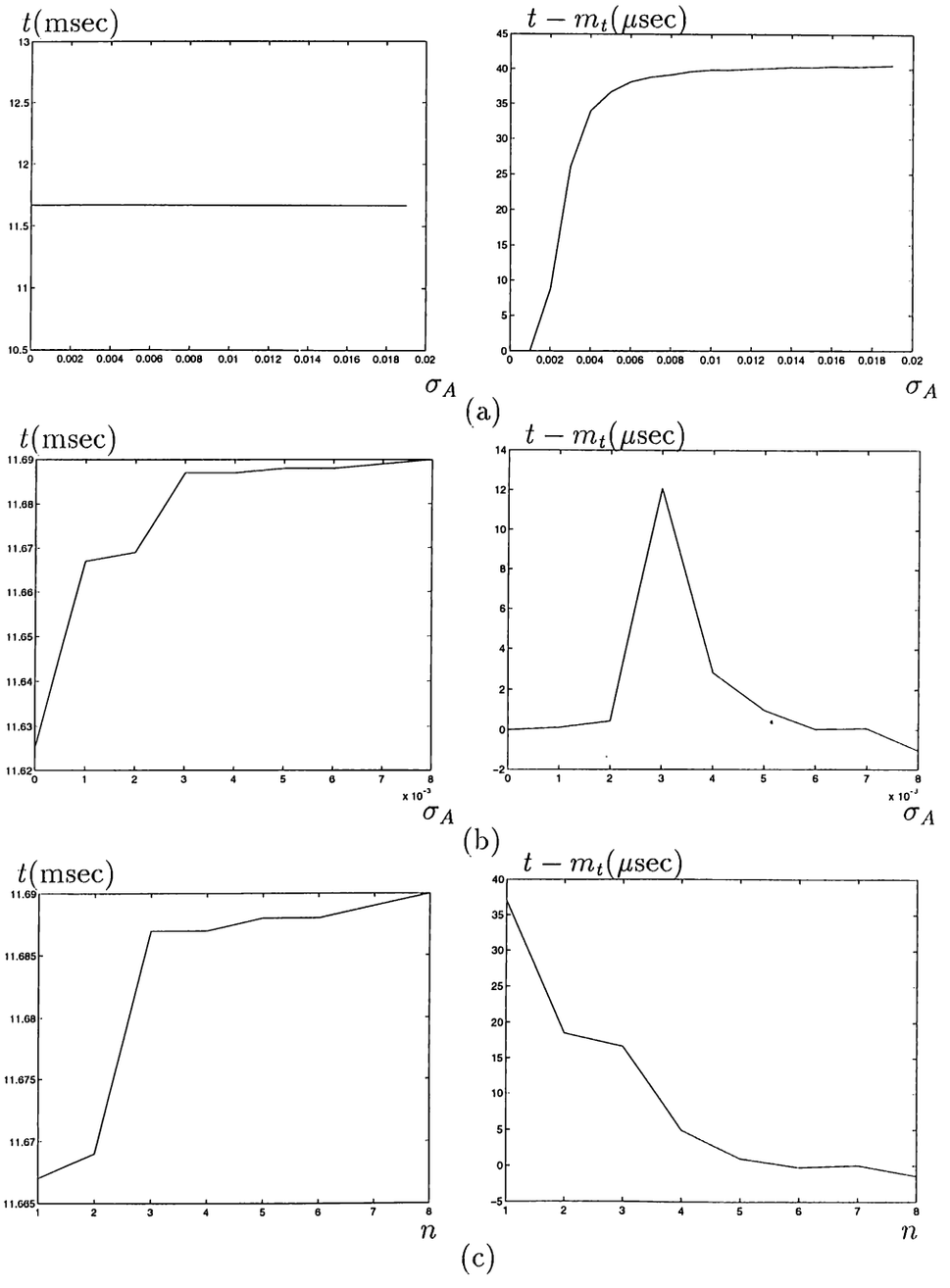


Figure E.3: True values (left column) and bias (right column) in the TOF measurements over 1000 realizations when a typical echo waveform is used with (a) $\tau = 0.005$ (b) $\tau = 5\sigma_A$ (c) $\tau = n\sigma_A$ and $\sigma_A = 0.005$.

E.2 Relationship between σ_A and σ_t in a Linear Signal Model

In the small interval in which the echo waveform first crosses the threshold level τ, the waveform can be modeled as a linear function of time as shown in Figure E.4. To keep the change in σ_t under control, a linear signal model of form $y(t) = at$ is investigated with different slope values a.

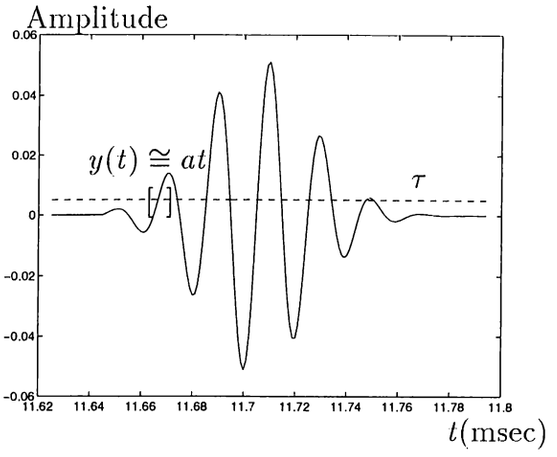


Figure E.4: A typical echo waveform and threshold level τ.

The analysis done in the previous section for a typical echo waveform to investigate the effect of change in amplitude noise standard deviation and threshold level is repeated here for the linear model with different slope values. The standard deviation and mean of TOF estimate and bias in the TOF measurements over 1000 realizations are given in Figures E.5 and E.6 respectively. In these figures, since the TOF's become smaller with increasing slope, σ_t, m_t and the bias decrease with increasing slope as expected. However, the bias in the TOF estimates increases with increasing σ_A independent of threshold level setting.

For a typical echo waveform, in the first cycle, the slope is of the order of hundreds, and in the second cycle it is of the order of thousands. The simulation results for $y(t) = 100t$ and $y(t) = 1000t$ are given in Figures E.7–E.10 separately. Referring to these figures, σ_t does not exceed 120 μsec for $a = 100$, and 10 μsec for $a = 1000$ for all noise standard deviations and threshold levels. Maximum bias in TOF estimates is around 220 μsec for $a = 100$ and 28 μsec for $a = 1000$. These standard deviations of TOF estimates and biases in TOF estimates are larger than the values of standard deviations and biases obtained for a typical echo waveform.

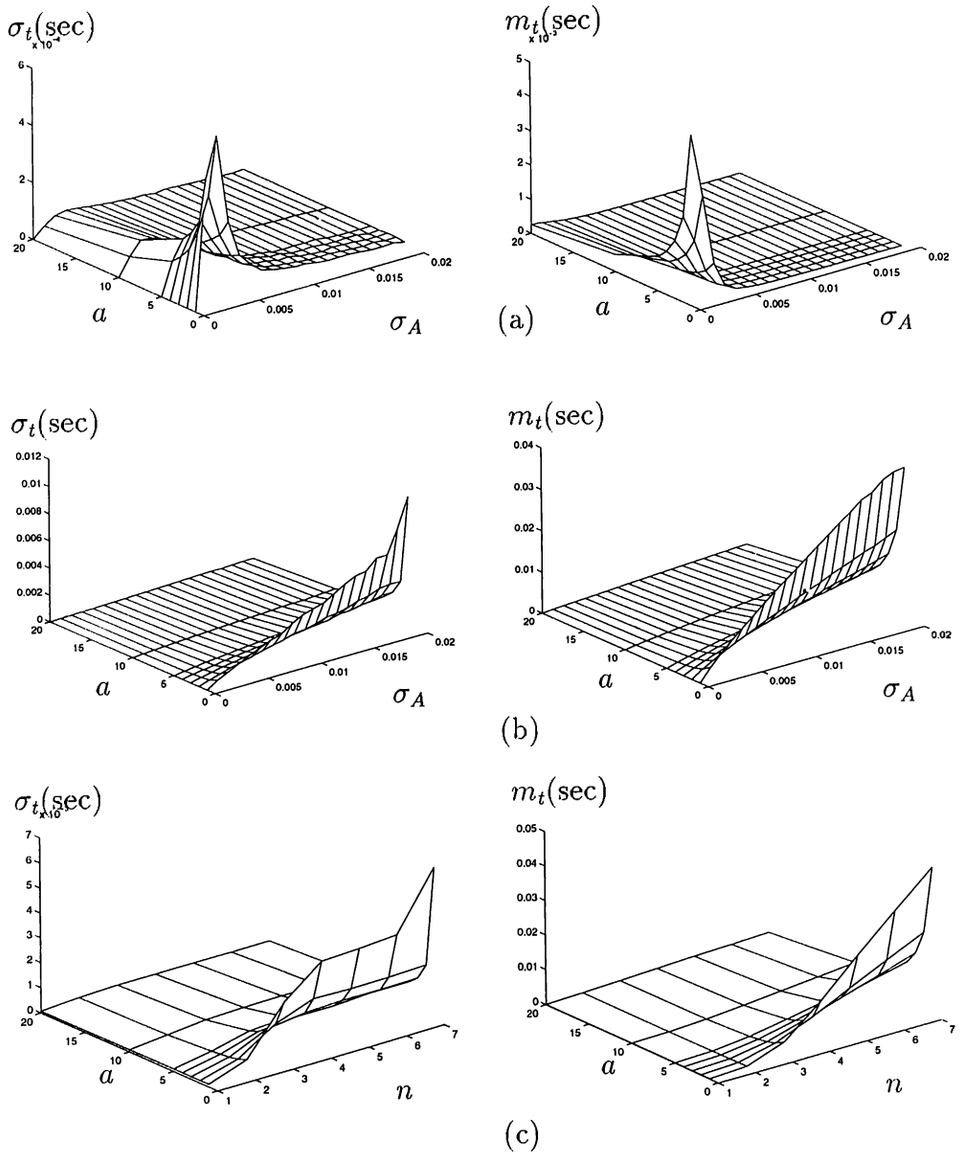


Figure E.5: Standard deviation and mean of TOF estimate over 1000 realizations when a linear signal model of form $y(t) = at$ is used with (a) $\tau = 0.005$ (b) $\tau = 5\sigma_A$ (c) $\tau = n\sigma_A$ and $\sigma_A = 0.01$.

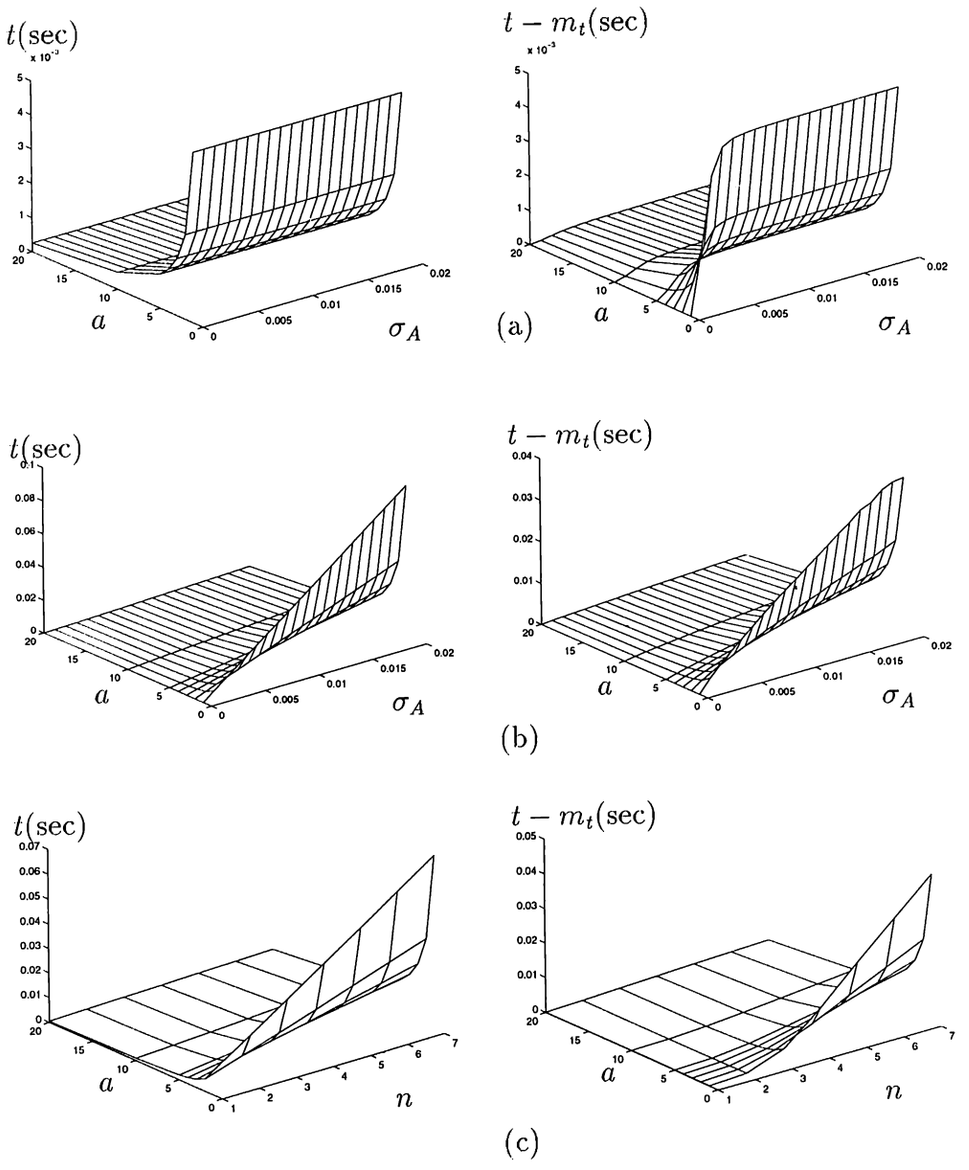


Figure E.6: True value (left column) and bias (right column) in the TOF measurements over 1000 realizations when a linear signal model of form $y(t) = at$ is used with (a) $\tau = 0.005$ (b) $\tau = 5\sigma_A$ (c) $\tau = n\sigma_A$ and $\sigma_A = 0.01$.

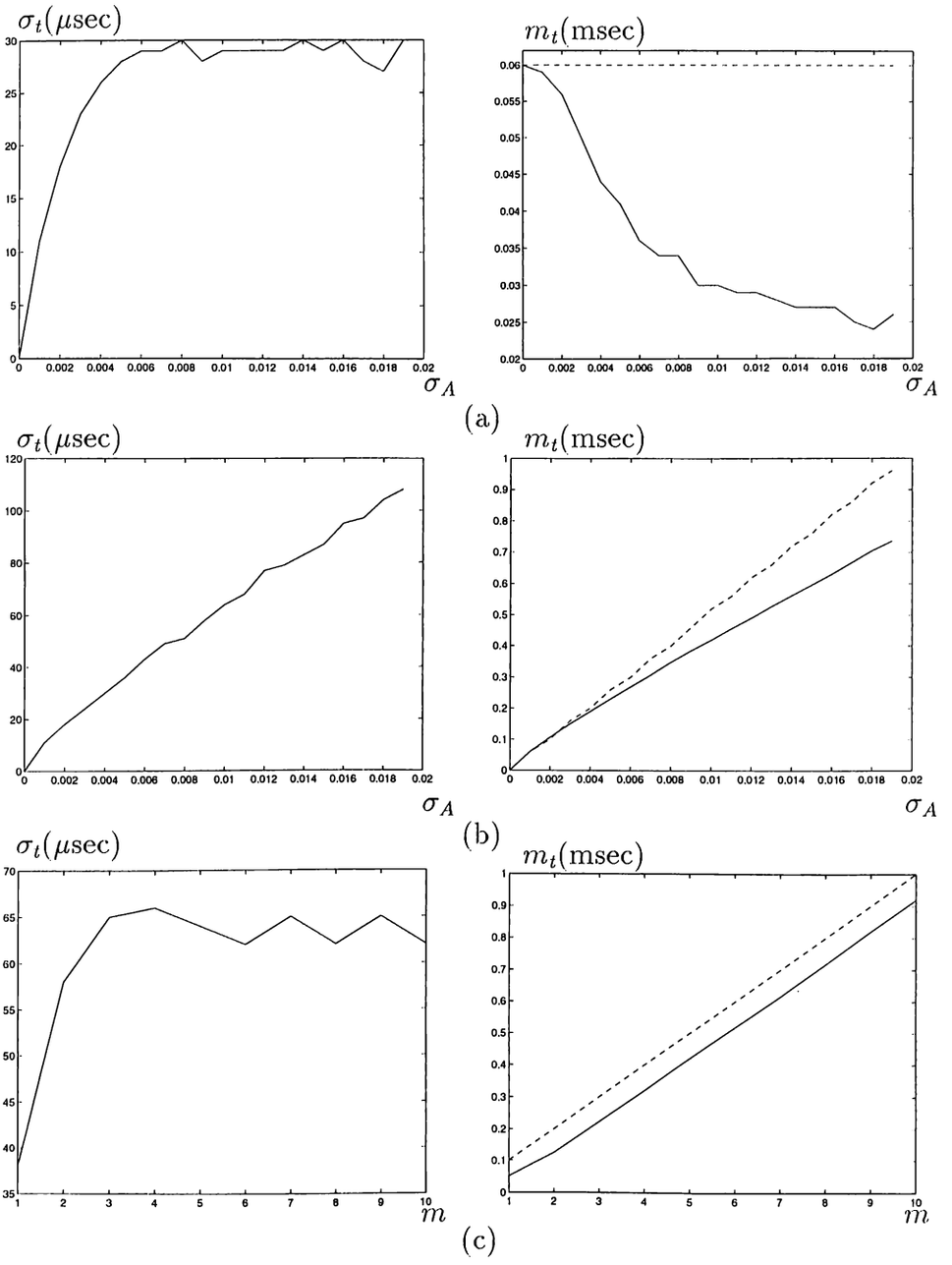


Figure E.7: Standard deviation and mean of TOF estimate over 1000 realizations when a linear signal model of form $y(t) = 100t$ is used with (a) $\tau = 0.005$ (b) $\tau = 5\sigma_A$ (c) $\tau = m\sigma_A$ and $\sigma_A = 0.01$.

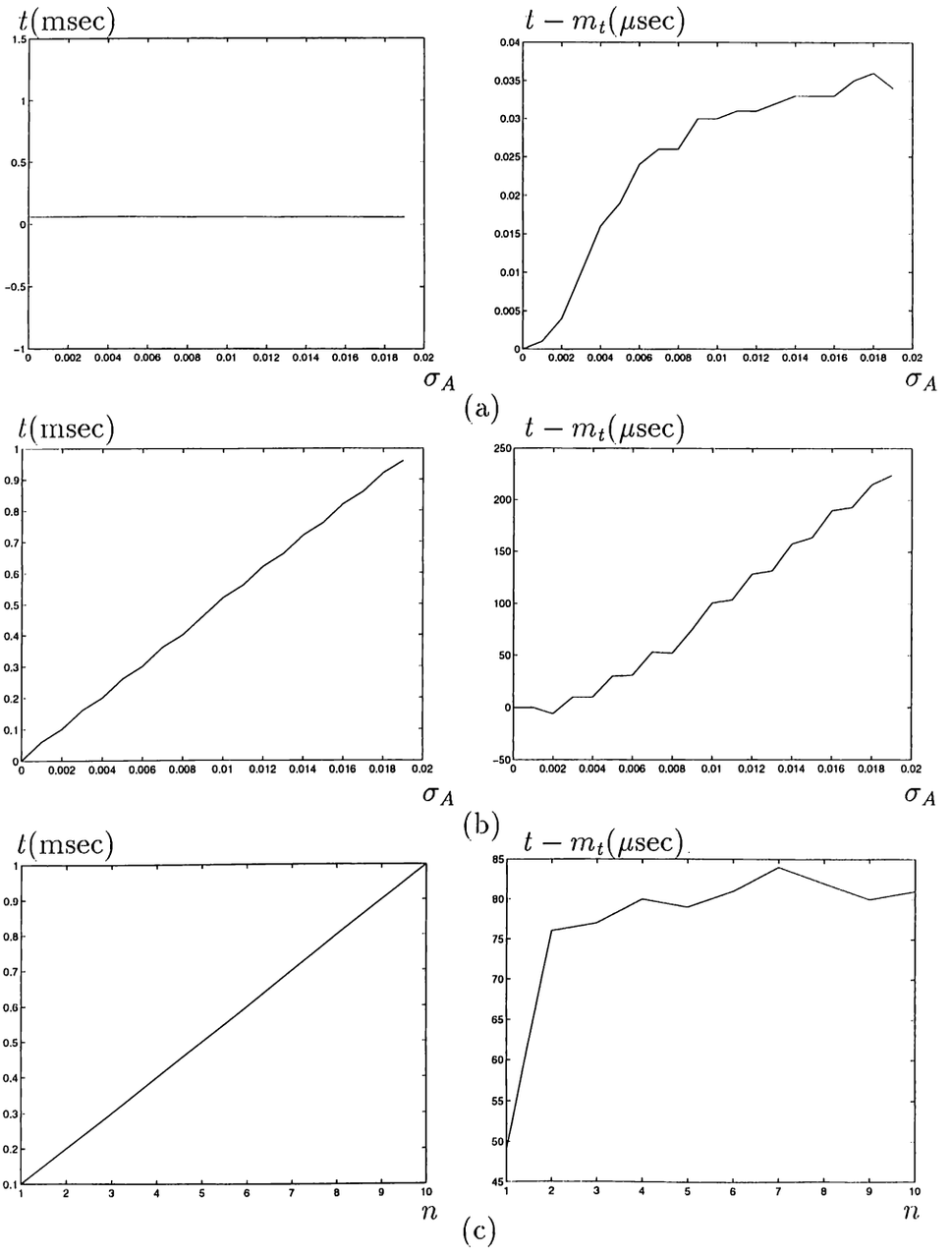


Figure E.8: True value (left column) and bias (right column) in the TOF measurements over 1000 realizations when a linear signal model of form $y(t) = 100t$ is used with (a) $\tau = 0.005$ (b) $\tau = 5\sigma_A$ (c) $\tau = n\sigma_A$ and $\sigma_A = 0.01$.

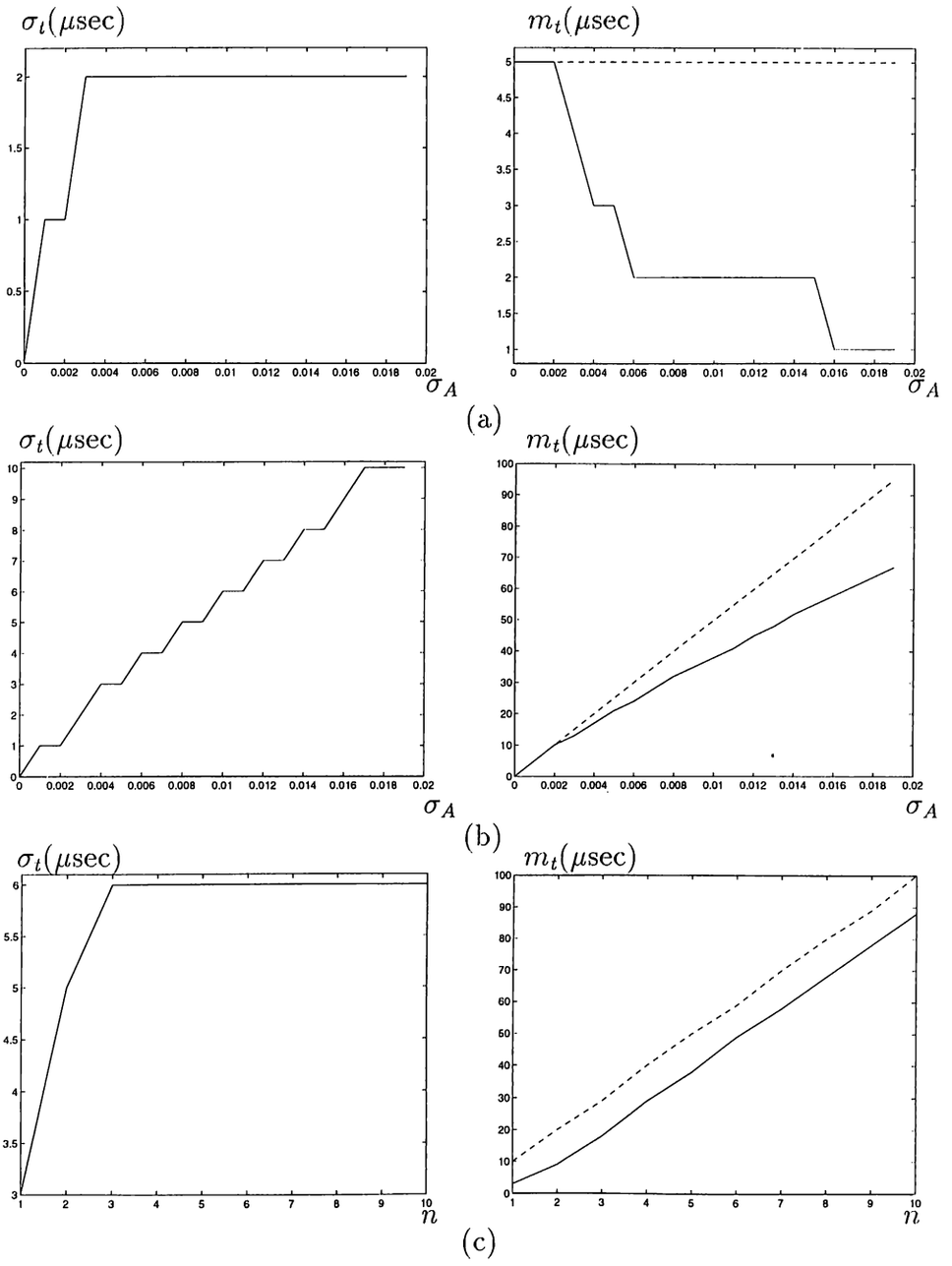


Figure E.9: Standard deviation and mean of TOF estimate over 1000 realizations when a linear signal model of form $y(t) = 1000t$ is used with (a) $\tau = 0.005$ (b) $\tau = 5\sigma_A$ (c) $\tau = n\sigma_A$ and $\sigma_A = 0.01$.

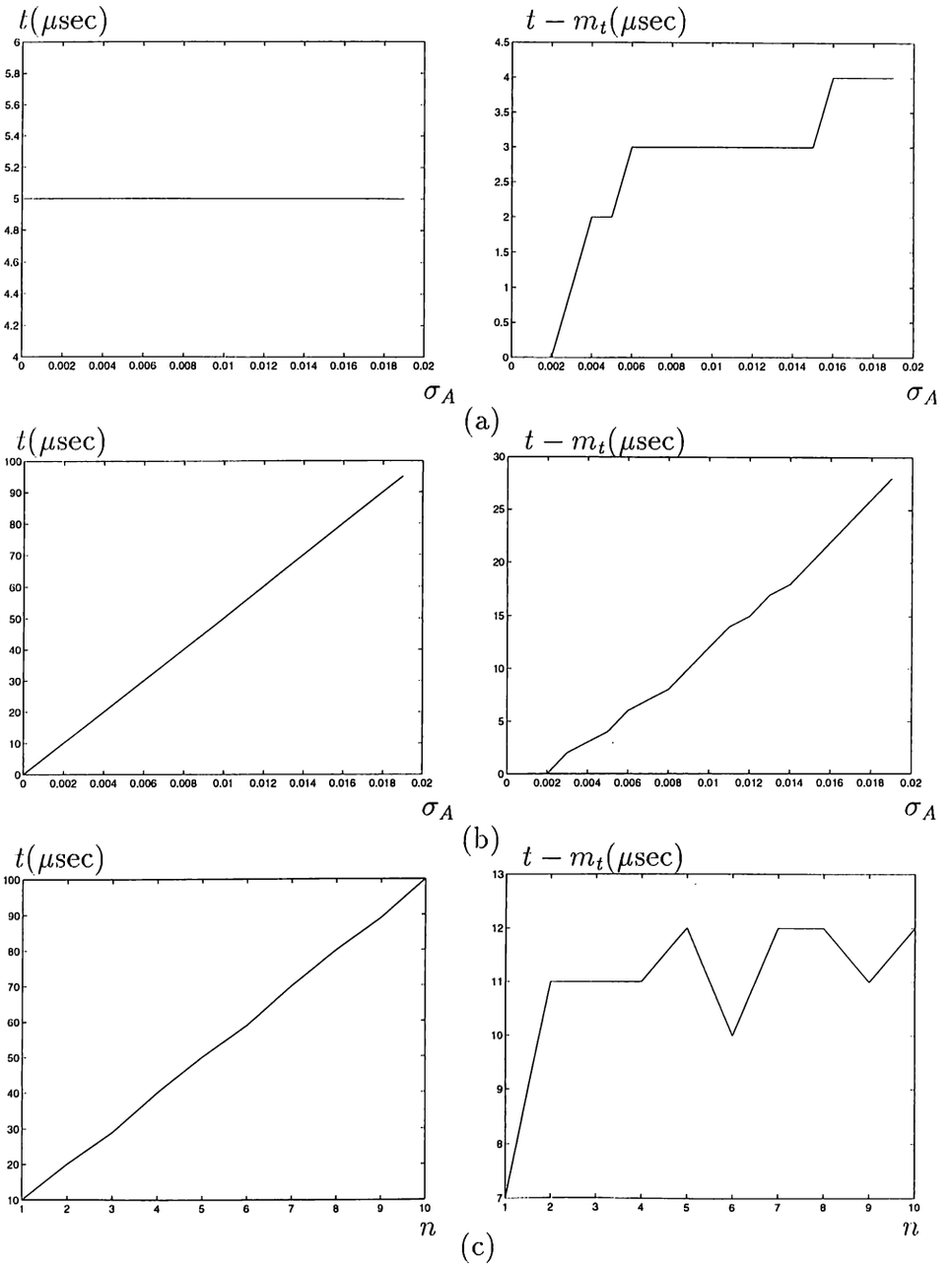


Figure E.10: True value (left column) and bias (right column) in the TOF measurements over 1000 realizations when a linear signal model of form $y(t) = 1000t$ is used with (a) $\tau = 0.005$ (b) $\tau = 5\sigma_A$ (c) $\tau = n\sigma_A$ and $\sigma_A = 0.01$.

Appendix F

COMPUTER PROGRAMS

In this appendix, the computer programs used in this study are provided. First ten programs are written in C++. Programs $1-5$ are used to produce the echo signal, amplitude characteristics and TOF characteristics of target primitives which are: plane, corner, edge, cylinder and acute corner respectively. The rest of the C++ programs are used to make belief assignments and feature fusion using Dempster-Shafer rule of combination. Eleventh program is written in PlaNet to find the weight values of the neural network using the backpropagation algorithm, and the last two are written in Matlab to find the output of the neural network.

Program 1: A C++ program to produce echo signal, amplitude and TOF characteristics of a planar target primitive:

```
# include <stdio.h>
# include <math.h>
# include <stdlib.h>

double f0=50000;
double Amax=1.028439;
double rmin=0.1;
double sigma=0.00003;
double sigmat=5;
double d=0.05;
double c=343.5;
double r=2.0;

double r1,r2,ra,rb,theta1,theta2,tofaa,tofab,tofba,tofbb;

// To calculate echo signal at time t:

double f(double t0,double tof,double t1,double t2,double r11,double r22)
{
double s;

s=(2*rmin*Amax/(r11+r22))*exp(-(t0-tof-3.0/f0)*(t0-tof-3.0/f0)/(sigma*sigma))
  *exp(-(t1*t1+t2*t2)/(2*sigmat*sigmat))*sin(2*M_PI*f0*(t0-tof));
return(s);}

// To calculate maximum echo amplitude at angle theta:

double f1(double r11,double r22,double t1,double t2)
{
double s;

s=(2*rmin*Amax/(r11+r22))*exp(-(t1*t1+t2*t2)/(2*sigmat*sigmat));
return(s);}

// To calculate range and time-of-flight values at angle theta:

void f2(double theta)
```



```
{
double th;

th=theta*M_PI/180.0;
ra=r-d*sin(th)/2.0;
rb=r+d*sin(th)/2.0;
r1=sqrt(r*r+d*d/4.0-r*d*sin(th));
r2=sqrt(r*r+d*d/4.0+r*d*sin(th));
theta1=90-(asin(r*cos(th)/r1))*180.0/M_PI;
theta2=90-(asin(r*cos(th)/r2))*180.0/M_PI;

tofaa=2*ra/c;
tofba=tofab=(r1+r2)/c;
tofbb=2*rb/c;
}

main()
{

double t,eaa,eba,ebb,eab,theta,Aaa,Aab,Aba,Abb;

int k,kfinal;
theta=0.0;

FILE *fout1;
FILE *fout2;
FILE *fout3;

fout1=fopen("echo.dat","w");
fout2=fopen("amplitude.dat","w");
fout3=fopen("TOF.dat","w");

if(tofaa <= tofbb)
 k=int (tofaa/(0.000001));
else
 k=int (tofbb/(0.000001));

kfinal=k+500;
k-=300;
```



```
f2(theta);

do
 {
   t=k*0.000001;
   eaa=f(t,tofaa,theta,theta,ra,ra);
   eab=f(t,tofba,theta1,theta2,r1,r2);
   eba=f(t,tofab,theta2,theta1,r2,r1);
   ebb=f(t,tofbb,theta,theta,rb,rb);
   fprintf(fout1,"%.15f %.15f %.15f %.15f %.15f \n",t,eaa,eab,eba,ebb);
   k++;
 }while(k<=kfinal);

theta=-15.0;

do
 {
   f2(theta);
   Aaa=f1(ra,ra,theta,theta);
   Aab=f1(r1,r2,theta1,theta2);
   Aba=f1(r2,r1,theta2,theta1);
   Abb=f1(rb,rb,theta,theta);
   fprintf(fout2,"%.15f %.15f %.15f %.15f %.15f \n",theta,Aaa,Aab,Aba,Abb);
   fprintf(fout3,"%.15f %.15f %.15f %.15f %.15f \n"
                                          ,theta,tofaa,tofab,tofba,tofbb);
   theta+=0.01;
 }while(theta<=15.0);

}
```

Program 2: A C++ program to produce echo signal, amplitude and TOF characteristics of a corner type target primitive:

```
# include <stdio.h>
# include <math.h>
# include <stdlib.h>

double f0=50000;
double Amax=1.070032;
double rmin=0.1;
```



```
double sigma=0.00003;
double sigmat=5;
double d=0.05;
double c=343.5;
double r=2.0;

double ra,rb,thetaa,thetab,tofaa,tofab,tofba,tofbb;

// To calculate echo signal at time t:

double f(double t0,double tof,double t1,double t2,double r11,double r22)
{
double s;

s=(2*rmin*Amax/(r11+r22))*exp(-(t0-tof-3.0/f0)*(t0-tof-3.0/f0)/(sigma*sigm
  *exp(-(t1*t1+t2*t2)/(2*sigmat*sigmat))*sin(2*M_PI*f0*(t0-tof));
return(s);}

// To calculate maximum echo amplitude at angle theta:

double f1(double r11,double r22,double t1,double t2)
{
double s;

s=(2*rmin*Amax/(r11+r22))*exp(-(t1*t1+t2*t2)/(2*sigmat*sigmat));
return(s);}

// To calculate range and time-of-flight values at angle theta:

void f2(double theta)
{
double th;

th=theta*M_PI/180.0;
ra=sqrt(r*r+d*d/4.0-r*d*sin(th));
rb=sqrt(r*r+d*d/4.0+r*d*sin(th));
thetaa=180.0*acos(r*cos(th)/ra)/M_PI;
thetab=180.0*acos(r*cos(th)/rb)/M_PI;

tofaa=2*ra/c;
```

```
tofba=tofab=2*r/c;
tofbb=2*rb/c;
}

main()
{

double t,eaa,eba,ebb,eab,theta,Aaa,Aab,Aba,Abb;

int k,kfinal;
theta=0.0;

FILE *fout1;
FILE *fout2;
FILE *fout3;

fout1=fopen("echo.dat","w");
fout2=fopen("amplitude.dat","w");
fout3=fopen("TOF.dat","w");

if(tofaa <= tofbb)
 k=int (tofaa/(0.000001));
else
 k=int (tofbb/(0.000001));

kfinal=k+500;
k-=300;

f2(theta);

do
 {
   t=k*0.000001;
   eaa=f(t,tofaa,thetaa,thetaa,ra,ra);
   eab=f(t,tofba,theta,theta,r,r);
   eba=f(t,tofab,theta,theta,r,r);
   ebb=f(t,tofbb,thetab,thetab,rb,rb);
   fprintf(fout1,"%.15f %.15f %.15f %.15f %.15f \n",t,eaa,eab,eba,ebb);
   k++;
 }while(k<=kfinal);
```

157

```
theta=-15.0;

do
 {
   f2(theta);
   Aaa=f1(ra,ra,thetaa,thetaa);
   Aab=f1(r,r,theta,theta);
   Aba=f1(r,r,theta,theta);
   Abb=f1(rb,rb,thetab,thetab);
   fprintf(fout2,"%.15f %.15f %.15f %.15f %.15f \n",theta,Aaa,Aab,Aba,Abb);
   fprintf(fout3,"%.15f %.15f %.15f %.15f %.15f \n"
                                          ,theta,tofaa,tofab,tofba,tofbb);
   theta+=0.01;
 }while(theta<=15.0);

}
```

Program 3: A C++ program to produce echo signal, amplitude and TOF characteristics of an edge type target primitive with a parameter θ_c:

```
# include <stdio.h>
# include <math.h>
# include <stdlib.h>

double f0=50000;
double Amax=1.168;
double rmin=0.1;
double sigma=0.00003;
double sigmat=5;
double d=0.05;
double c=343.5;
double r=2.0;
double thetae=90.0;

double r1,r2,theta1,theta2,tofaa,tofab,tofba,tofbb,roe;

// To calculate echo signal at time t:

double f(double t0,double tof,double t1,double t2,double r11,double r22)
```

158

```
{
double s;

s=(roe*Amax*pow(rmin,1.5)/(r11*sqrt(r22)))
  *exp(-(t0-tof-3.0/f0)*(t0-tof-3.0/f0)/(sigma*sigma))
  *exp(-(t1*t1+t2*t2)/(2*sigmat*sigmat))
  *sin(2*M_PI*f0*(t0-tof));
return(s);}

// To calculate maximum echo amplitude at angle theta:

double f1(double r11,double r22,double t1,double t2)
{
double s;

s=(roe*Amax*pow(rmin,1.5)/(r11*sqrt(r22)))
  *exp(-(t1*t1+t2*t2)/(2*sigmat*sigmat));
return(s);}

// To calculate range and time-of-flight values at angle theta:

void f2(double theta)
{
double th;

th=theta*M_PI/180.0;
r1=sqrt(r*r+d*d/4.0-r*d*sin(th));
r2=sqrt(r*r+d*d/4.0+r*d*sin(th));
theta1=90.0-(asin(r*cos(th)/r1))*180.0/M_PI;
theta2=90.0-(asin(r*cos(th)/r2))*180.0/M_PI;

tofaa=2*r1/c;
tofba=tofba=(r1+r2)/c;
tofbb=2*r2/c;
}

main()
{

double t,eaa,eba,ebb,eab,theta,Aaa,Aab,Aba,Abb;
```

159

```
int k,kfinal;
theta=0.0;

FILE *fout1;
FILE *fout2;
FILE *fout3;

fout1=fopen("echo.dat","w");
fout2=fopen("amplitude.dat","w");
fout3=fopen("TOF.dat","w");

if(tofaa <= tofbb)
 k=int (tofaa/(0.000001));
else
 k=int (tofbb/(0.000001));

kfinal=k+500;
k-=300;

roe=0.48316*thetae*M_PI/180.0-0.25298;
f2(theta);

do
 {
   t=k*0.000001;
   eaa=f(t,tofaa,theta1,theta1,r1,r1);
   eab=f(t,tofba,theta1,theta2,r1,r2);
   eba=f(t,tofab,theta2,theta1,r2,r1);
   ebb=f(t,tofbb,theta2,theta2,r2,r2);
   fprintf(fout1,"%.15f %.15f %.15f %.15f %.15f \n",t,eaa,eab,eba,ebb);
   k++;
 }while(k<=kfinal);

theta=-15.0;

do
 {
   f2(theta);
   Aaa=f1(r1,r1,theta1,theta1);
```

160

```
   Aab=f1(r1,r2,theta1,theta2);
   Aba=f1(r2,r1,theta2,theta1);
   Abb=f1(r2,r2,theta2,theta2);
   fprintf(fout2,"%.15f %.15f %.15f %.15f %.15f \n",theta,Aaa,Aab,Aba,Abb);
   fprintf(fout3,"%.15f %.15f %.15f %.15f %.15f \n"
                                          ,theta,tofaa,tofab,tofba,tofbb);

   theta+=0.01;
 }while(theta<=15.0);

}
```

Program 4: A C++ program to produce echo signal, amplitude and TOF characteristics of a cylindrical target primitive with a parameter r_c:

```
# include <stdio.h>
# include <math.h>
# include <stdlib.h>

double f0=50000;
double Amax=6.8887;
double rmin=0.1;
double sigma=0.00003;
double sigmat=5;
double d=0.05;
double c=343.5;
double r=2.0;
double rc=0.2;

double r1,r2,ra,rb,theta1,theta2,thetaa,thetab,tofaa,tofab,tofba,tofbb,roc;

// To calculate echo signal at time t:

double f(double t0,double tof,double t1,double t2,double r11,double r22)
{
double s;

s=(roc*Amax*pow(rmin,1.5)/(r11*sqrt(r22)))
  *exp(-(t0-tof-3.0/f0)*(t0-tof-3.0/f0)/(sigma*sigma))
  *exp(-(t1*t1+t2*t2)/(2*sigmat*sigmat))
  *sin(2*M_PI*f0*(t0-tof));
```



```
return(s);}

// To calculate maximum echo amplitude at angle theta:

double f1(double r11,double r22,double t1,double t2)
{
double s;

s=(roc*Amax*pow(rmin,1.5)/(r11*sqrt(r22)))
  *exp(-(t1*t1+t2*t2)/(2*sigmat*sigmat));
return(s);}

// To calculate range and time-of-flight values at angle theta:

void f2(double theta)
{
double th;

th=theta*M_PI/180.0;

r1=sqrt(r*r+d*d/4.0-r*d*sin(th));
r2=sqrt(r*r+d*d/4.0+r*d*sin(th));
ra=sqrt(pow(r+rc,2.0)+d*d/4.0-d*(r+rc)*sin(th))-rc;
rb=sqrt(pow(r+rc,2.0)+d*d/4.0+d*(r+rc)*sin(th))-rc;

theta1=90.0-180.0*asin(r*cos(th)/r1)/M_PI;
theta2=90.0-180.0*asin(r*cos(th)/r2)/M_PI;
thetaa=90.0-180.0*asin((r+rc)*cos(th)/(ra+rc))/M_PI;
thetab=90.0-180.0*asin((r+rc)*cos(th)/(rb+rc))/M_PI;

tofaa=2*ra/c;
tofba=(r1+r2)/c;
tofbb=2*rb/c;
}

main()
{

double t,eaa,eba,ebb,eab,theta,Aaa,Aab,Aba,Abb;
```



```
int k,kfinal;
theta=0.0;

FILE *fout1;
FILE *fout2;
FILE *fout3;

fout1=fopen("echo.dat","w");
fout2=fopen("amplitude.dat","w");
fout3=fopen("TOF.dat","w");

if(tofaa <= tofbb)
 k=int (tofaa/(0.000001));
else
 k=int (tofbb/(0.000001));

kfinal=k+500;
k-=300;

roc=0.44946*rc-0.022471;
f2(theta);

do
 {
   t=k*0.000001;
   eaa=f(t,tofaa,thetaa,thetaa,ra,ra);
   eab=f(t,tofba,theta1,theta2,r1,r2);
   eba=f(t,tofab,theta2,theta1,r2,r1);
   ebb=f(t,tofbb,thetab,thetab,rb,rb);
   fprintf(fout1,"%.15f %.15f %.15f %.15f %.15f \n",t,eaa,eab,eba,ebb);
   k++;
 }while(k<=kfinal);

theta=-15.0;

do
 {
   f2(theta);
   Aaa=f1(ra,ra,thetaa,thetaa);
   Aab=f1(r1,r2,theta1,theta2);
```



```
   Aba=f1(r2,r1,theta2,theta1);
   Abb=f1(rb,rb,thetab,thetab);
   fprintf(fout2,"%.15f %.15f %.15f %.15f %.15f \n",theta,Aaa,Aab,Aba,Abb);
   fprintf(fout3,"%.15f %.15f %.15f %.15f %.15f \n"
                                          ,theta,tofaa,tofab,tofba,tofbb);

   theta+=0.01;
 }while(theta<=15.0);

}
```

Program 5: A C++ program to produce echo signal, amplitude and TOF characteristics of an acute corner type target primitive with a parameter θ_c:

```
# include <stdio.h>
# include <math.h>
# include <stdlib.h>

double f0=50000;
double Amax=451.8753;
double rmin=0.1;
double sigma=0.00003;
double sigmat=5;
double d=0.05;
double c=343.5;
double r=2.0;
double theta0=12.0;
double thetac=45.0;

double ra,rb,thetaa,thetab,tofaa,tofab,thetaf,theta1f,tofba,tofbb,thetat,
       theta1,theta2,raa,rbb,rab,rba,beta,alfaa,alfab,betaa,betab,alfaab,
       betaab,alfaba,betaba;

// To calculate echo signal at time t:

double f(double t0,double tof,double t1,double t2,double r11)
{
double s;

s=(2*Amax*rmin/r11)*exp(-(t1*t1+t2*t2)/(2*sigmat*sigmat))
  *exp(-(t0-tof-3/f0)*(t0-tof-3/f0)/(sigma*sigma))
```

```
  *sin(2*M_PI*f0*(t0-tof));
return(s);}

// To calculate maximum echo amplitude at angle theta:

double f1(double r11,double t1,double t2)
{
double s;

s=(2*Amax*rmin/r11)*exp(-(t1*t1+t2*t2)/(2.0*sigmat*sigmat));
return(s);}

// To find angle theta2:

double findtheta()
{

double error,errorf,a,b,thetak,thetaf;

errorf=1.0;
thetaf=1.0;

do
 {
  ra=sqrt(r*r+d*d/4.0-r*d*sin(thetak*M_PI/180.0));
  rb=sqrt(r*r+d*d/4.0+r*d*sin(thetak*M_PI/180.0));
  thetat=180*acos((ra*ra+rb*rb-d*d)/(2.0*ra*rb))/M_PI;
  beta=(thetac-thetat)/2.0;
  a=(pow(sin((beta)*M_PI/180.0),2.0)
    -pow(sin(2*theta0*M_PI/180.0),2.0))/(pow(sin((180-beta-2.0*theta0)
    *M_PI/180.0),2.0))+1;
  b=2.0*sin((beta)*M_PI/180.0)*cos(2.0*theta0*M_PI/180.0)
    /sin((180-beta-2.0*theta0)*M_PI/180.0);
  error=a-b;

 if(fabs(error)<=errorf)
  {
   errorf=fabs(error);
   thetaf=thetak;
  }
```



```
 thetak+=0.1;
 }while(thetak>=1 && thetak<=60)
return(thetaf);
}

// To calculate range and time-of-flight values at angle theta:

void f2(double theta)
{
double th,thc,thp,s1,rp,r1,r2,thetap;

th=theta*M_PI/180.0;
thc=thetac*M_PI/180.0;
ra=sqrt(r*r+d*d/4.0-r*d*sin(th));
rb=sqrt(r*r+d*d/4.0+r*d*sin(th));
thetat=180*acos((ra*ra+rb*rb-d*d)/(2.0*ra*rb))/M_PI;
beta=(thetac-thetat)/2.0;
thetaa=180.0*acos(r*cos(th)/ra)/M_PI;
thetab=180.0*acos(r*cos(th)/rb)/M_PI;
raa=2*ra*sin(thc);
rbb=2*rb*sin(thc);
s1=ra*ra+rb*rb-d*d;
theta1=90-thetac/2.0;
theta2=findtheta();

if(theta == 0.0)
 {
   rab=rba=sqrt(ra*ra+rb*rb-2*ra*rb*cos(M_PI*(2*thetac-thetat)/180));
   alfaa=alfab=90-thetac+thetaa;
   betaa=betab=90-thetac-thetaa;
   alfaab=alfaba=90-thetac-thetaa+thetat/2;
   betaab=betaba=90-thetac-thetaa+thetat/2;
 }
else if(0.0 < theta && theta < theta2)
 {
   rab=rba=sqrt(ra*ra+rb*rb-2*ra*rb*cos(M_PI*(2*thetac-thetat)/180));
   alfaa=90-thetac-thetaa;
   betaa=90-thetac+thetaa;
   alfab=90-thetac+thetab;
   betab=90-thetac-thetab;
```



```
   alfaab=180*acos((ra*ra+rab*rab-rb*rb)/(2*ra*rab))/M_PI+thetaa;
   betaab=180*acos((rb*rb+rab*rab-ra*ra)/(2*rb*rab))/M_PI-thetab;
   alfaba=180*acos((-ra*ra+rba*rba+rb*rb)/(2*rb*rba))/M_PI-thetab;
   betaba=180*acos((ra*ra+rba*rba-rb*rb)/(2*ra*rba))/M_PI+thetaa;
 }
else if((theta1-theta0) > theta && theta > theta2)
 {
   rab=sqrt(ra*ra+rb*rb-2.0*ra*rb*cos(M_PI*(2*thetac-thetat)/180));
   rba=sqrt(ra*ra+rb*rb-2.0*ra*rb*cos(M_PI*(2*thetac+thetat)/180));
   alfaa=90-thetac+thetaa;
   betaa=90-thetac-thetaa;
   alfab=90-thetac+thetab;
   betab=90-thetac-thetab;
   alfaab=180.0*acos((rab*rab+d*d-rbb*rbb)/(2*rab*d))/M_PI-90;
   betaab=180.0*acos((rab*rab-raa*raa+d*d)/(2*d*rab))/M_PI-90;
   alfaba=90-180.0*acos((rba*rba-raa*raa+d*d)/(2*d*rba))/M_PI;
   betaba=90-180.0*acos((rba*rba-rbb*rbb+d*d)/(2*d*rba))/M_PI;
 }
else if(-1.0*theta2 < theta && theta < 0.0)
 {
   rab=rba=sqrt(ra*ra+rb*rb-2*ra*rb*cos(M_PI*(2*thetac-thetat)/180));
   alfaa=90-thetac+thetaa;
   betaa=90-thetac-thetaa;
   alfab=90-thetac-thetab;
   betab=90-thetac+thetab;
   alfaab=180*acos((ra*ra+rab*rab-rb*rb)/(2*ra*rab))/M_PI-thetaa;
   betaab=180*acos((rb*rb+rab*rab-ra*ra)/(2*rb*rab))/M_PI+thetab;
   alfaba=180*acos((-ra*ra+rba*rba+rb*rb)/(2*rb*rba))/M_PI+thetab;
   betaba=180*acos((ra*ra+rba*rba-rb*rb)/(2*ra*rba))/M_PI-thetaa;
 }
else if(-1*(theta1-theta0) < theta && theta < -1.0*theta2)
 {
   rab=sqrt(ra*ra+rb*rb-2*ra*rb*cos(M_PI*(2*thetac+thetat)/180));
   rba=sqrt(ra*ra+rb*rb-2*ra*rb*cos(M_PI*(2*thetac-thetat)/180));
   alfaa=90-thetac+thetaa;
   betaa=90-thetac-thetaa;
   alfab=90-thetac+thetab;
   betab=90-thetac-thetab;
   alfaab=90-180.0*acos((rab*rab+d*d-rbb*rbb)/(2*rab*d))/M_PI;
   betaab=90-180.0*acos((rab*rab-raa*raa+d*d)/(2*d*rab))/M_PI;
```



```
   alfaba=180.0*acos((rba*rba-raa*raa+d*d)/(2*d*rba))/M_PI-90;
   betaba=180.0*acos((rba*rba-rbb*rbb+d*d)/(2*d*rba))/M_PI-90;
 }
else if((theta1-theta0) <= theta && theta <= (theta1+theta0) ||
        -1.0*(theta1+theta0) <= theta && theta <= -1.0*(theta1-theta0))
 {
   rp=r*sin(thc/2);
   if(theta < 0.0)
    thp=(theta1+theta)*M_PI/180;
   else
    thp=M_PI*theta1/180-th;
   ra=rp-d*sin(thp)/2.0;
   rb=rp+d*sin(thp)/2.0;
   r1=sqrt(rp*rp+d*d/4.0-rp*d*sin(thp));
   r2=sqrt(rp*rp+d*d/4.0+rp*d*sin(thp));
   raa=2*ra;
   rbb=2.0*rb;
   rab=rba=r1+r2;
   alfaba=betaab=90-(asin(rp*cos(thp)/r1))*180.0/M_PI;
   alfaab=betaba=90-(asin(rp*cos(thp)/r2))*180.0/M_PI;
   alfaa=betaa=alfab=betab=thetap;
 }

tofaa=raa/c;
tofbb=rbb/c;
tofba=rba/c;
tofab=rab/c;

}

main()
{

double t,eaa,eba,ebb,eab,theta,Aaa,Aab,Aba,Abb;

int k,kfinal;
theta=0.0;

FILE *fout1;
FILE *fout2;
```

```
FILE *fout3;

fout1=fopen("echo.dat","w");
fout2=fopen("amplitude.dat","w");
fout3=fopen("TOF.dat","w");

if(tofaa <= tofbb)
 k=int (tofaa/(0.000001));
else
 k=int (tofbb/(0.000001));

kfinal=k+500;
k-=300;

f2(theta);

do
 {
   t=k*0.000001;
   eaa=f(t,tofaa,alfaa,betaa,raa);
   eab=f(t,tofba,alfaab,betaab,rab);
   eba=f(t,tofab,alfaba,betaba,rba);
   ebb=f(t,tofbb,alfab,betab,rbb);
   fprintf(fout1,"%.15f %.15f %.15f %.15f %.15f \n",t,eaa,eab,eba,ebb);
   k++;
 }while(k<=kfinal);

theta=-15.0;

do
 {
   f2(theta);
   Aaa=f1(raa,alfaa,betab);
   Aab=f1(rab,alfaab,betaab);
   Aba=f1(rba,alfaba,betaba);
   Abb=f1(rbb,alfab,betab);
   fprintf(fout2,"%.15f %.15f %.15f %.15f %.15f \n",theta,Aaa,Aab,Aba,Abb);
   fprintf(fout3,"%.15f %.15f %.15f %.15f %.15f \n"
                                        ,theta,tofaa,tofab,tofba,tofbb);
   theta+=0.01;
```



```
 }while(theta<=15.0);

}
```

Program 6: A C++ program to assign belief values to plane and corner for a single sensor pair in the simulated room:

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <stddef.h>
#include <string.h>

double N( );
void set_seed( );
double f1( );
void plane( );
void corner( );
double signal( );
void getdata( );
double findpmax( );
double findcmax( );
void findtarget( );

double maxaaa,maxab,maxba,maxbb,ntofaa,ntofab,ntofba,ntofbb,xtarget,
       ytarget,bap,bac,br,bth,a,b,bau,fplane,fcorner,funknown,pmax,
       cmax,vmaxaaa,vmaxbb,vmaxab,vmaxba,vtofaa,vtofab,vtofba,vtofbb;

double rmax=8.0;
double thetamax=15.0;

double f0=50000;
double sigmat=5.0;
double sigma=0.00003;
double Amax=1.0;
double d=0.05;
double rmin=0.1;

double stdn=pow(10.0,-7.0);
int M1=801;
```



```
double c=343.5;
double aa[801],ab[801],bb[801],ba[801],timet[801];

// To compute the magnitude of echo signal at time t:

double f1(double t0,double tof,double t1,double t2,double r11,double r22)
{
double s;

s=(2*rmin*Amax/(r11+r22))*exp(-(t0-tof)*(t0-tof)/(sigma*sigma))
   *exp(-(t1*t1+t2*t2)/(2*sigmat*sigmat))*sin(2*M_PI*f0*(t0-tof));
return(s);}

// To produce echo signal which is reflected from a plane at range r
// and inclination angle theta:

void plane(double r,double theta)
{

double r1,r2,theta1,theta2,tofaa,tofba,tofbb,t,ra,rb,th,tofab;

int k,kfinal,m;
Amax=1.028075;
th=theta*M_PI/180.0;
r1=r-d*sin(th)/2.0;
r2=r+d*sin(th)/2.0;
ra=sqrt(r*r+d*d/4.0-r*d*sin(th));
rb=sqrt(r*r+d*d/4.0+r*d*sin(th));
theta1=90-(asin(r*cos(th)/ra))*180.0/M_PI;
theta2=90-(asin(r*cos(th)/rb))*180.0/M_PI;
tofaa=2*r1/c;
tofba=tofab=(ra+rb)/c;
tofbb=2*r2/c;

if(tofaa <= tofbb)
k=int (tofaa/(0.000001));
else
k=int (tofbb/(0.000001));

kfinal=k+500;
```



```
k-=300;
m=0;
do
{
 t=k*0.000001;
 timet[m]=t;
 aa[m]=f1(t,tofaa,theta,theta,r1,r1);
 ba[m]=f1(t,tofba,theta1,theta2,ra,rb);
 bb[m]=f1(t,tofbb,theta,theta,r2,r2);
 ab[m]=f1(t,tofab,theta2,theta1,rb,ra);
 m++;
 k++;
}while(k <= kfinal);

}

// To produce echo signal which is reflected from a corner at range r
// and inclination angle theta:

void  corner(double r, double theta)
{

double thetaa,thetab,tofaa,tofab,tofba,tofbb,t,ra,rb,th;
int k,kfinal,m;

Amax=1.070032;

th=theta*M_PI/180.0;
ra=sqrt(r*r+d*d/4.0-r*d*sin(th));
rb=sqrt(r*r+d*d/4.0+r*d*sin(th));
thetaa=180.0*acos(r*cos(th)/ra)/M_PI;
thetab=180.0*acos(r*cos(th)/rb)/M_PI;

tofaa=2*ra/c;
tofab=tofba=2*r/c;
tofbb=2*rb/c;

if(tofaa <= tofbb)
 k=int (tofaa/(0.000001));
else
```

```
 k=int (tofbb/(0.000001));

kfinal=k+500;
k-=300;
m=0;

do
{
 t=k*0.000001;
 timet[m]=t;
 aa[m]=f1(t,tofaa,thetaa,thetaa,ra,ra);
 ba[m]=f1(t,tofba,theta,theta,r,r);
 ab[m]=f1(t,tofab,theta,theta,r,r);
 bb[m]=f1(t,tofbb,thetab,thetab,rb,rb);
 m++;
 k++;
}while(k <= kfinal);

}

// To produce the echo signals when the sensor pair scans the simulated room:

double signal(double xs,double ys,double phi)
{
/*
   (xs,ys) : coordinates of the sensor pair;
   phi     : angle between horizontal axis and sensor pair whose
             range -180 to 180; */

double r0,thetap,xt,yt,t,flag;
int k,m;

xt=5.0; /* horizontal length of the room */
yt=5.0; /* vertical length of the room */
flag=0.0;

if(-13.0 < phi && phi < 13.0 )
{
//Target is a plane.
 xt/=2.0;
```



```
 yt=0.0;
 r0=xt-xs;
 thetap=phi;
 plane(r0,thetap);
 flag=1.0;

}
else if(30.0 < phi && phi < 60.0)
{
//Target is a corner.
 xt/=2.0;
 yt/=2.0;
 r0=sqrt((xt-xs)*(xt-xs)+(yt-ys)*(yt-ys));
 thetap=phi-180.0*atan2((yt-ys),(xt-xs))/M_PI;
 corner(r0,thetap);
 flag=1.0;

}
else if(77.0 < phi && phi < 103.0)
{
//Target is a plane.
 xt=0.0;
 yt/=2.0;
 r0=yt-ys;
 thetap=phi-90.0;
 plane(r0,thetap);
 flag=1.0;

}
else if(120.0 < phi && phi < 150.0)
{
//Target is a corner.
 xt/=-2.0;
 yt/=2.0;
 r0=sqrt((xt-xs)*(xt-xs)+(yt-ys)*(yt-ys));
 thetap=phi-180.0*atan2((yt-ys),(xt-xs))/M_PI;
 corner(r0,thetap);
 flag=1.0;

}
```



```
else if( 167.0 <phi && phi <= 180.0 || -180.0 <= phi && phi < -167)
{
//Target is a plane.
 xt/=-2.0;
 yt=0.0;
 r0=fabs(xt-xs);
 if( phi > 0.0 )
  thetap=phi-180;
 else
  thetap=phi+180;
 plane(r0,thetap);
 flag=1.0;

}
else if(-150.0 < phi && phi < -120.0)
{
//Target is a corner.
 xt/=-2.0;
 yt/=-2.0;
 r0=sqrt((xt-xs)*(xt-xs)+(yt-ys)*(yt-ys));
 thetap=phi-180.0*atan2((yt-ys),(xt-xs))/M_PI;
 corner(r0,thetap);
 flag=1.0;

}
else if( -103.0 < phi && phi < -77.0)
{
//Target is a plane.
 xt=0.0;
 yt/=-2.0;
 r0=fabs(yt-ys);
 thetap=phi+90.0;
 plane(r0,thetap);
 flag=1.0;

}
else if ( -60.0 < phi && phi < -30.0 )
{
//Target is a corner.
 xt/=2.0;
```



```
 yt/=-2.0;
 r0=sqrt((xt-xs)*(xt-xs)+(yt-ys)*(yt-ys));
 thetap=phi-180.0*atan2((yt-ys),(xt-xs))/M_PI;
 corner(r0,thetap);
 flag=1.0;

}

else
{
flag=0.0;
}
return(flag);
}

// To get the maximum amplitude and the TOF of the echo signal:

double getdata(void)
{

double trh,flag;
int k;

flag=1.0;
trh=5*stdn;
vmaxaaa=0.0;
vmaxab=0.0;
vmaxba=0.0;
vmaxbb=0.0;

// To find the maximum amplitude of echo waveforms:

for(k=i;k < (M1-2);k++)
{
 if (fabs(ab[k]) > vmaxab)
 vmaxab=fabs(ab[k]);
 if (fabs(ba[k]) > vmaxba)
 vmaxba=fabs(ba[k]);
 if (fabs(bb[k]) > vmaxbb)
 vmaxbb=fabs(bb[k]);
```

```
 if (fabs(aa[k]) > vmaxaaa)
 vmaxaaa=fabs(aa[k]);

}

// To find the TOF of echo waveforms:

k=0;
do
 k++;
while(aa[k] < trh && k<=M1-2);
vtofaa=timet[k];
if(k==M1-1)
flag=0.0;

k=0;
do
 k++;
while(ba[k] < trh && k<=M1-2);
vtofba=timet[k];
if(k==M1-1)
flag=0.0;

k=0;
do
 k++;
while(ab[k] < trh && k<=M1-2);
vtofab=timet[k];
if(k==M1-1)
flag=0.0;

k=0;
do
 k++;
while(bb[k] < trh  && k<=M1-2);
vtofbb=timet[k];
if(k==M1-1)
flag=0.0;

return(flag);
```



```
}

// To find the scaling factor in the assigned belief value to a plane:

double findpmax(double r)
{
plane(r,0.0);
getdata();
pmax=(vmaxaaa-vmaxab);
return(pmax);
}

// To find the scaling factor in the assigned belief value to a corner:

double findcmax(double r)
{
double angle,diff;
angle=-15.0;
cmax=0.0;
do
{
 corner(r,angle);
 getdata();
 diff=fabs(vmaxab-vmaxaaa)+fabs(vmaxab-vmaxbb);
 if(diff > cmax)
  cmax=diff;
 angle++;
}while(angle <= 15.0);
return(cmax);
}

// Classification of the target primitives:

void findtarget(double xsp, double ysp, double phis)
{
/*
   (xsp,ysp) : coordinates of the sensor pair;
   phip      : angle between horizontal axis and sensor pair whose
               range -180 to 180; */
```



```
double raa,rab,rbb,r,theta;

if(maxbb > 5*stdn)
{
if(maxaaa > 5*stdn)
{
if(maxab > 5*stdn)
{

fplane = fcorner = funknown = 0.0;

// Plane-corner differentiation:

if((maxaaa-maxab) > 1.0*stdn && (maxbb-maxab) >1.0*stdn)
fplane=1.0;
else if((maxab-maxaaa) > 1.0*stdn || (maxab-maxbb) > 1.0*stdn)
fcorner=1.0;
else
funknown = 1.0;

raa=ntofaa*c/2.0;
rab=ntofab*c;
rbb=ntofbb*c/2.0;

// Belief assignment to the plane and estimation of position of this plane:

if(fplane==1.0)
{
 r=(raa+rbb)/2.0;
 theta=180.0*asin((rbb-raa)/d)/M_PI;
 findpmax(r);
 bap=(maxaaa-maxab)+(maxbb-maxab)/(2*pmax);
 bac=0.0;
 bau=1.0-bap;
 br=(rmax-r)/(rmax-rmin);
 bth=(thetamax-fabs(theta))/thetamax;
 if( -45.0 <= phis && phis < 45.0)
 {
  xtarget=xsp+r;
  ytarget=ysp;
```



```
 }
 else if( 45.0 <= phis && phis <135)
 {
  xtarget=xsp;
  ytarget=ysp+r;
 }
 else if( 135.0 <= phis && phis <180.0 || -180.0 <= phis && phis <-135.0)
 {
  xtarget=xsp-r;
  ytarget=ysp;
 }
 else if(-135.0 <= phis  && phis < -45.0)
 {
  xtarget=xsp;
  ytarget=ysp-r;
 }
}

// Belief assignment to the corner and estimation of position of this plar

if (fcorner==1.0)
{
 r=rab/2.0;
 theta=180.0*asin((rbb*rbb-raa*raa)/(2.0*r*d))/M_PI;
 findcmax(r);
 bac=(fabs(maxab-maxaaa)+fabs(maxab-maxbb))/cmax;
 bap=0.0;
 bau=1.0-bac;
 br=(rmax-r)/(rmax-rmin);
 bth=(thetamax-fabs(theta))/thetamax;
 if( 0.0 <= phis && phis < 90.0)
 {
  xtarget=xsp+r*cos(M_PI*(phis-theta)/180.0);
  ytarget=ysp+r*sin(M_PI*(phis-theta)/180.0);
 }
 else if( 90.0 <= phis && phis <= 180.0)
 {
  xtarget=xsp-r*cos(M_PI*(180-(phis-theta))/180.0);
  ytarget=ysp+r*sin(M_PI*(180-(phis-theta))/180.0);
 }
```

```
 else if( -180 <= phis && phis < -90.0)
 {
  xtarget=xsp-r*cos(M_PI*(180.0-(theta-phis))/180.0);
  ytarget=ysp-r*sin(M_PI*(180.0-(theta-phis))/180.0);
 }
 else if( -90.0 <= phis && phis <0.0)
 {
  xtarget=xsp+r*cos(M_PI*(theta-phis)/180.0);
  ytarget=ysp-r*sin(M_PI*(theta-phis)/180.0);
 }
}

// Belief assignment to an unknown target:

if(funknown==1.0)
{
bap=bac=br=bth=xtarget=ytarget=0.0;
bau=1.0;
}
}
else
{
bap=bac=br=bth=xtarget=ytarget=0.0;
bau=1.0;
}
}
else
{
bap=bac=br=bth=xtarget=ytarget=0.0;
bau=1.0;
}
}
else
{
bap=bac=br=bth=xtarget=ytarget=0.0;
bau=1.0;
}
}

main()
```



```
{

double xpos,ypos,phi,control,control1;
int m;

FILE *fout1;
fout1=fopen("belief.dat","w");

// Position of the sensor pair:

xpos=0.0;
ypos=0.0;

phi=-180.0;
do
{
 control=signal(xpos,ypos,phi);
 if( control !=0.0)
 {
  control1=getdata();
  if(control1 !=0.0)
  {
   maxaaa=vmaxaaa;
   maxbb=vmaxbb;
   maxab=vmaxab;
   maxba=vmaxba;
   ntofaa=vtofaa;
   ntofab=vtofab;
   ntofba=vtofba;
   ntofbb=vtofbb;
   findtarget(xpos,ypos,phi);
  }
 }
 // To print the belief values at angle phi in an output file in the form
 // angle phi, x coordinate of target (xtarget), y coordinate of target
 // (ytarget), b(plane), b(corner), b(unknown), b(r) and b(theta):

 fprintf(fout1," %.15f %.15f %.15f %.15f %.15f %.15f %.15f %.15f \n",
              phi,xtarget,ytarget,bap,bac,bau,br,bth);
 phi++;
```



```
}while(phi<=180.0);

}
```

Program 7: A C++ program to find the classification results of a single sensor pair in the simulated room:

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <stddef.h>
#include <string.h>
#include <time.h>
#define SWAP(a,b) {float temp=(a);(a)=(b);(b)=temp;}
#define Random_Seed 0           /* 0 = randomize */
#define RAND_MAX (2147483647)

double N( );
void set_seed( );
double f1( );
void plane( );
void corner( );
double signal( );
double getdata( );
double findpmax( );
double findcmax( );
void findtarget( );
void getstatistics( );

double maxaaa,maxab,maxba,maxbb,ntofaa,ntofab,ntofba,ntofbb,xtarget,
       ytarget,bap,bac,br,bth,a,b,bau,fplane,fcorner,funknown,pmax,
       cmax,vmaxaaa,vmaxbb,vmaxab,vmaxba,vtofaa,vtofab,vtofba,vtofbb,
       stdn,countt,countf,countu,probt,probf,probu,Amax,count;

double rmax=8.0;
double thetamax=15.0;

double f0=50000.0;
double sigmat=5.0;
double sigma=0.00003;
```



```
double d=0.05;
double rmin=0.1;

int M1=801;
double c=343.5;
double aa[801],ab[801],bb[801],ba[801],timet[801];

// To produce noise:

double N(double m, double std)
/* calculates two normal (mean m, variance std**2) values, returns one */
{
/*if (!Random_Seed) set_seed(); else srand(Random_Seed);*/
/* ornegin z+= N(0.,std_noise); diye kullanilabilir */

   double fac, u1, u2, v1, v2, s;
   int u;
   static int iset = 0;
   static double gset;
   if (iset == 0)
   {
      s = 0;
      do
      {
 u = rand ();
 u1 = (double) u / RAND_MAX;
 u = rand ();
 u2 = (double) u / RAND_MAX;
 v1 = 2 * u1 - 1;
 v2 = 2 * u2 - 1;
 s = v1 * v1 + v2 * v2;
      } while (s > 1);
      fac = sqrt ((-2 * log (s)) / s);
      gset = (v1 * fac);
      iset = 1;
      v2 = v2 * fac * std + m;
      return v2;
   }
   else
   {
```

```
      iset = 0;
      return (gset * std) + m;
   }
}

void set_seed (void)
/* routine to set seed of random number generator in a 'random' fashion
   by using the time */
{
   srand ((unsigned int) time(NULL));
}

// To compute the magnitude of echo signal at time t:

double f1(double t,double tof,double t1,double t2,double r11,double r22)
{
double s;

s=(2*rmin*Amax/(r11+r22))*exp(-(t-tof)*(t-tof)/(sigma*sigma))
  *exp(-(t1*t1+t2*t2)/(2*sigmat*sigmat))*sin(2*M_PI*f0*(t-tof));
return(s);}

// To produce echo signal which is reflected from a plane at range r
// and inclination angle theta:

void plane(double r,double theta)
{

double r1,r2,theta1,theta2,tofaa,tofba,tofbb,t,ra,rb,th,tofab;

int k,kfinal,m;
Amax=1.028075;
th=theta*M_PI/180.0;
r1=r-d*sin(th)/2.0;
r2=r+d*sin(th)/2.0;
ra=sqrt(r*r+d*d/4.0-r*d*sin(th));
rb=sqrt(r*r+d*d/4.0+r*d*sin(th));
theta1=90-(asin(r*cos(th)/ra))*180.0/M_PI;
theta2=90-(asin(r*cos(th)/rb))*180.0/M_PI;
tofaa=2*r1/c;
```



```
tofba=tofab=(ra+rb)/c;
tofbb=2*r2/c;

if(tofaa <= tofbb)
k=int (tofaa/(0.000001));
else
k=int (tofbb/(0.000001));

kfinal=k+500;
k-=300;
m=0;
do
{
 t=k*0.000001;
 timet[m]=t;
 aa[m]=f1(t,tofaa,theta,theta,r1,r1);
 ba[m]=f1(t,tofba,theta1,theta2,ra,rb);
 bb[m]=f1(t,tofbb,theta,theta,r2,r2);
 ab[m]=f1(t,tofab,theta2,theta1,rb,ra);
 m++;
 k++;
}while(k <= kfinal);

}

// To produce echo signal which is reflected from a corner at range r
// and inclination angle theta:

void  corner(double r, double theta)
{

double thetaa,thetab,tofaa,tofab,tofba,tofbb,t,ra,rb,th;
int k,kfinal,m;

Amax=1.070032;

th=theta*M_PI/180.0;
ra=sqrt(r*r+d*d/4.0-r*d*sin(th));
rb=sqrt(r*r+d*d/4.0+r*d*sin(th));
thetaa=180.0*acos(r*cos(th)/ra)/M_PI;
```



```
thetab=180.0*acos(r*cos(th)/rb)/M_PI;

tofaa=2*ra/c;
tofab=tofba=2*r/c;
tofbb=2*rb/c;

if(tofaa <= tofbb)
 k=int (tofaa/(0.000001));
else
 k=int (tofbb/(0.000001));

kfinal=k+500;
k-=300;
m=0;

do
{
 t=k*0.000001;
 timet[m]=t;
 aa[m]=f1(t,tofaa,thetaa,thetaa,ra,ra);
 ba[m]=f1(t,tofba,theta,theta,r,r);
 ab[m]=f1(t,tofab,theta,theta,r,r);
 bb[m]=f1(t,tofbb,thetab,thetab,rb,rb);
 m++;
 k++;
}while(k <= kfinal);

}

// To produce the echo signals when the sensor pair scans the simulated
// room:

double signal(double xs,double ys,double phi)
{
/*
   (xs,ys) : coordinates of the sensor pair;
   phi     : angle between horizontal axis and sensor pair whose
             range -180 to 180; */

double r0,thetap,xt,yt,t,flag;
```



```
int k,m;

xt=5.0; /* horizontal length of the room */
yt=5.0; /* vertical length of the room */
flag=0.0;

if(-13.0 < phi && phi < 13.0 )
{
//Target is a plane.
 xt/=2.0;
 yt=0.0;
 r0=xt-xs;
 thetap=phi;
 plane(r0,thetap);
 flag=1.0;

}
else if(30.0 < phi && phi < 60.0)
{
//Target is a corner.
 xt/=2.0;
 yt/=2.0;
 r0=sqrt((xt-xs)*(xt-xs)+(yt-ys)*(yt-ys));
 thetap=phi-180.0*atan2((yt-ys),(xt-xs))/M_PI;
 corner(r0,thetap);
 flag=1.0;

}
else if(77.0 < phi && phi < 103.0)
{
//Target is a plane.
 xt=0.0;
 yt/=2.0;
 r0=yt-ys;
 thetap=phi-90.0;
 plane(r0,thetap);
 flag=1.0;

}
else if(120.0 < phi && phi < 150.0)
```

```
{
//Target is a corner.
 xt/=-2.0;
 yt/=2.0;
 r0=sqrt((xt-xs)*(xt-xs)+(yt-ys)*(yt-ys));
 thetap=phi-180.0*atan2((yt-ys),(xt-xs))/M_PI;
 corner(r0,thetap);
 flag=1.0;

}
else if( 167.0 <phi && phi <= 180.0 || -180.0 <= phi && phi < -167)
{
//Target is a plane.
 xt/=-2.0;
 yt=0.0;
 r0=fabs(xt-xs);
 if( phi > 0.0 )
  thetap=phi-180;
 else
  thetap=phi+180;
 plane(r0,thetap);
 flag=1.0;

}
else if(-150.0 < phi && phi < -120.0)
{
//Target is a corner.
 xt/=-2.0;
 yt/=-2.0;
 r0=sqrt((xt-xs)*(xt-xs)+(yt-ys)*(yt-ys));
 thetap=phi-180.0*atan2((yt-ys),(xt-xs))/M_PI;
 corner(r0,thetap);
 flag=1.0;

}
else if( -103.0 < phi && phi < -77.0)
{
//Target is a plane.
 xt=0.0;
 yt/=-2.0;
```



```
 r0=fabs(yt-ys);
 thetap=phi+90.0;
 plane(r0,thetap);
 flag=1.0;

}
else if ( -60.0 < phi && phi < -30.0 )
{
//Target is a corner.
 xt/=2.0;
 yt/=-2.0;
 r0=sqrt((xt-xs)*(xt-xs)+(yt-ys)*(yt-ys));
 thetap=phi-180.0*atan2((yt-ys),(xt-xs))/M_PI;
 corner(r0,thetap);
 flag=1.0;

}

else
{
flag=0.0;
}
return(flag);
}

// To get the maximum amplitude and the TOF of the echo signal:

double getdata(void)
{

double trh,flag;
int k;

flag=1.0;
trh=5*stdn;
vmaxaaa=0.0;
vmaxab=0.0;
vmaxba=0.0;
vmaxbb=0.0;
```



```
// To add the Gaussian noise with mean 0 and standard deviation stdn
// on the echo waveforms:

for(k=1;k < (M1-2);k++)
{
 ab[k]+=N(0.0,stdn);
 bb[k]+=N(0.0,stdn);
 aa[k]+=N(0.0,stdn);
 ba[k]+=N(0.0,stdn);
}

// To find the maximum amplitude of echo waveforms:

for(k=1;k < (M1-2);k++)
{
 if (fabs(ab[k]) > vmaxab)
 vmaxab=fabs(ab[k]);
 if (fabs(ba[k]) > vmaxba)
 vmaxba=fabs(ba[k]);
 if (fabs(bb[k]) > vmaxbb)
 vmaxbb=fabs(bb[k]);
 if (fabs(aa[k]) > vmaxaaa)
 {
 vmaxaaa=fabs(aa[k]);
 }

}

// To find the TOF of echo waveforms:

k=0;
do
 k++;
while(aa[k] < trh && k<=M1-2);
vtofaa=timet[k];
if(k==M1-1)
flag=0.0;

k=0;
do
```



```
 k++;
while(ba[k] < trh && k<=M1-2);
vtofba=timet[k];
if(k==M1-1)
flag=0.0;

k=0;
do
 k++;
while(ab[k] < trh && k<=M1-2);
vtofab=timet[k];
if(k==M1-1)
flag=0.0;

k=0;
do
 k++;
while(bb[k] < trh  && k<=M1-2);
vtofbb=timet[k];
if(k==M1-1)
flag=0.0;

return(flag);
}

// To find the scaling factor in the assigned belief value to a plane:

double findpmax(double r)
{
plane(r,0.0);
getdata();
pmax=(vmaxaaa-vmaxab);
return(pmax);
}

// To find the scaling factor in the assigned belief value to a corner:

double findcmax(double r)
{
double angle,diff;
```

```
angle=-15.0;
cmax=0.0;
do
{
 corner(r,angle);
 getdata();
 diff=fabs(vmaxab-vmaxaaa)+fabs(vmaxab-vmaxbb);
 if(diff > cmax)
  cmax=diff;
 angle++;
}while(angle <= 15.0);
return(cmax);
}

// Classification of the target primitives:

void findtarget(double xsp, double ysp, double phis)
{
/*
   (xsp,ysp) : coordinates of the sensor pair;
   phip      : angle between horizontal axis and sensor pair whose
               range -180 to 180; */

double raa,rab,rbb,r,theta;

if(maxbb > 5*stdn)
{
if(maxaaa > 5*stdn)
{
if(maxab > 5*stdn)
{

fplane = fcorner = funknown = 0.0;

// Plane-corner differentiation:

if((maxaaa-maxab) > 1.0*stdn && (maxbb-maxab) >1.0*stdn)
fplane=1.0;
else if((maxab-maxaaa) > 1.0*stdn || (maxab-maxbb) > 1.0*stdn)
fcorner=1.0;
```



```
// Belief assignment to the corner and estimation of position of this plane:

if (fcorner==1.0)
{
 r=rab/2.0;
 theta=180.0*asin((rbb*rbb-raa*raa)/(2.0*r*d))/M_PI;
 findcmax(r);
 bac=(fabs(maxab-maxaaa)+fabs(maxab-maxbb))/cmax;
 bap=0.0;
 bau=1.0-bac;
 br=(rmax-r)/(rmax-rmin);
 bth=(thetamax-fabs(theta))/thetamax;
 if( 0.0 <= phis && phis < 90.0)
 {
  xtarget=xsp+r*cos(M_PI*(phis-theta)/180.0);
  ytarget=ysp+r*sin(M_PI*(phis-theta)/180.0);
 }
 else if( 90.0 <= phis && phis <= 180.0)
 {
  xtarget=xsp-r*cos(M_PI*(180-(phis-theta))/180.0);
  ytarget=ysp+r*sin(M_PI*(180-(phis-theta))/180.0);
 }
 else if( -180 <= phis && phis < -90.0)
 {
  xtarget=xsp-r*cos(M_PI*(180.0-(theta-phis))/180.0);
  ytarget=ysp-r*sin(M_PI*(180.0-(theta-phis))/180.0);
 }
 else if( -90.0 <= phis && phis <0.0)
 {
  xtarget=xsp+r*cos(M_PI*(theta-phis)/180.0);
  ytarget=ysp-r*sin(M_PI*(theta-phis)/180.0);
 }
}

// Belief assignment to an unknown target:

if(funknown==1.0)
{
bap=bac=br=bth=xtarget=ytarget=0.0;
```



```
else
funknown = 1.0;

raa=ntofaa*c/2.0;
rab=ntofab*c;
rbb=ntofbb*c/2.0;

// Belief assignment to the plane and estimation of position of this plane

if(fplane==1.0)
{
 r=(raa+rbb)/2.0;
 theta=180.0*asin((rbb-raa)/d)/M_PI;
 findpmax(r);
 bap=(maxaaa-maxab)+(maxbb-maxab)/(2*pmax);
 bac=0.0;
 bau=1.0-bap;
 br=(rmax-r)/(rmax-rmin);
 bth=(thetamax-fabs(theta))/thetamax;
 if( -45.0 <= phis && phis < 45.0)
 {
  xtarget=xsp+r;
  ytarget=ysp;
 }
 else if( 45.0 <= phis && phis <135)
 {
  xtarget=xsp;
  ytarget=ysp+r;
 }
 else if( 135.0 <= phis && phis <180.0 || -180.0 <= phis && phis <-135.0)
 {
  xtarget=xsp-r;
  ytarget=ysp;
 }
 else if(-135.0 <= phis  && phis < -45.0)
 {
  xtarget=xsp;
  ytarget=ysp-r;
 }
}
```



```
bau=1.0;
}
}
else
{
bap=bac=br=bth=xtarget=ytarget=0.0;
bau=1.0;
}
}
else
{
bap=bac=br=bth=xtarget=ytarget=0.0;
bau=1.0;
}
}
else
{
bap=bac=br=bth=xtarget=ytarget=0.0;
bau=1.0;
}
}

// To obtain the probability of correct classification, misclassification
// and unknown:

void getstatistics(double phi)
{

if(-13.0 < phi && phi < 13.0 || 77.0 < phi && phi < 103.0 || 167.0 <phi &&
 phi <= 180.0 || -180.0 <= phi && phi < -167 || -103.0 < phi && phi < -77.
{
if(bap >bac)
{
countt+=bap;
count+=bap;
}
else if(bac>bap)
{
countf+=bac;
count+=bac;
```

```
}
else
{
countu+=bau;
count+=bau;
}
}
else if(-150.0 < phi && phi < -120.0 || -60.0 < phi && phi < -30.0 ||
         120.0 < phi && phi < 150.0 || 30.0 < phi && phi < 60.0 )
{
if(bap >bac)
{
countf+=bap;
count+=bap;
}
else if(bac>bap )
{
countt+=bac;
count+=bac;
}
else
{
countu+=bau;
count+=bau;
}
}
else
{

if(bap >bac )
{
countf+=bap;
count+=bap;
}
else if(bac>bap)
{
countf+=bac;
count+=bac;
}
else
```



```
{
countu+=bau;
count+=bau;
}

}
}


main()
{
double xpos,ypos,phi,control,control1;
int m,k;

FILE *fout1;
fout1=fopen("probability.dat","w");

// Sensor pair position:

xpos=0.0;
ypos=0.0;

stdn=0.0;

count=0.0;
countt=0.0;
countf=0.0;
countu=0.0;

if (!Random_Seed)
                set_seed();
        else
                srand(Random_Seed);

do
{
 for(k=1;k<=5;k++)
 {
  phi=-180.0;
  do
```



```
  {
   control=signal(xpos,ypos,phi);
   if( control !=0.0)
   {
    control1=getdata();
    if(control1 !=0.0)
    {
     maxaaa=vmaxaaa;
     maxbb=vmaxbb;
     maxab=vmaxab;
     maxba=vmaxba;
     ntofaa=vtofaa;
     ntofab=vtofab;
     ntofba=vtofba;
     ntofbb=vtofbb;
     findtarget(xpos,ypos,phi);
     getstatistics(phi);
    }
   }
   phi++;
  }while(phi<180.0);
 }

 probt=(countt/count);
 probf=(countf/count);
 probu=(countu/count);

 countt=0.0;
 countf=0.0;
 countu=0.0;
 count=0.0;

 // To print the probabilities to an output file in the form of: amplitude
 // noise standard deviation (stdn), probability of correct classification
 // (probt), probability of misclassification (probf), probability of
 // unknown (probu).

 fprintf(fout1," %.15f %.15f %.15f %.15f  \n",stdn,probt,probf,probu);
 stdn+=0.001;
}while(stdn <= 0.018);
```



```
}
```

Program 8: A C++ program for feature fusion from three sensor pairs in the simulated room:

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <stddef.h>
#include <string.h>
#include <time.h>
#define SWAP(a,b) {float temp=(a);(a)=(b);(b)=temp;}
#define Random_Seed 0           /* 0 = randomize */
#define RAND_MAX (2147483647)

double N( );
void set_seed( );
double f1( );
void plane( );
void corner( );
double signal( );
double getdata( );
double findpmax( );
double findcmax( );
void findtarget( );
void getstatistics( );

double maxaaa,maxab,maxba,maxbb,ntofaa,ntofab,ntofba,ntofbb,xtarget,
       ytarget,bap,bac,br,bth,a,b,bau,fplane,fcorner,funknown,pmax,
       cmax,vmaxaaa,vmaxbb,vmaxab,vmaxba,vtofaa,vtofab,vtofba,vtofbb,
       stdn,countt,countf,countu,probt,probf,probu,Amax,rf,thetaf,
       count,rtr,thetat;

double rmax=8.0;
double thetamax=15.0;

double f0=50000;
double sigmat=5.0;
double sigma=0.00003;
```

```
double d=0.05;
double rmin=0.1;

int M1=801;
double c=343.5;
double aa[801],ab[801],bb[801],ba[801],timet[801];

// to generate Gaussian white noise:

double N(double m, double std)
/* calculates two normal (mean m, variance std**2) values, returns one */
{
/*if (!Random_Seed) set_seed(); else srand(Random_Seed);*/
/* ornegin z+= N(0.,std_noise); diye kullanilabilir */

   double fac, u1, u2, v1, v2, s;
   int u;
   static int iset = 0;
   static double gset;
   if (iset == 0)
   {
      s = 0;
      do
      {
 u = rand ();
 u1 = (double) u / RAND_MAX;
 u = rand ();
 u2 = (double) u / RAND_MAX;
 v1 = 2 * u1 - 1;
 v2 = 2 * u2 - 1;
 s = v1 * v1 + v2 * v2;
      } while (s > 1);
      fac = sqrt ((-2 * log (s)) / s);
      gset = (v1 * fac);
      iset = 1;
      v2 = v2 * fac * std + m;
      return v2;
   }
   else
   {
```



```
      iset = 0;
      return (gset * std) + m;
   }
}

void set_seed (void)
/* routine to set seed of random number generator in a 'random'
   fashion by using the time */
{
   srand ((unsigned int) time(NULL));
}

// To calculate magnitude of echo signal at time t:

double f1(double t0,double tof,double t1,double t2,double r11,double r22)
{
double s;

s=(2*rmin*Amax/(r11+r22))*exp(-(t0-tof)*(t0-tof)/(sigma*sigma))
  *exp(-(t1*t1+t2*t2)/(2*sigmat*sigmat))*sin(2*M_PI*f0*(t0-tof));
return(s);}

// To produce echo signal which is reflected from a plane at range r
// and inclination angle theta:

void plane(double r,double theta)
{

double r1,r2,theta1,theta2,tofaa,tofba,tofbb,t,ra,rb,th,tofab;
int k,kfinal,m;

Amax=1.028075;
th=theta*M_PI/180.0;
r1=r-d*sin(th)/2.0;
r2=r+d*sin(th)/2.0;
ra=sqrt(r*r+d*d/4.0-r*d*sin(th));
rb=sqrt(r*r+d*d/4.0+r*d*sin(th));
theta1=90-(asin(r*cos(th)/ra))*180.0/M_PI;
theta2=90-(asin(r*cos(th)/rb))*180.0/M_PI;
```



```
tofaa=2*r1/c;
tofba=tofab=(ra+rb)/c;
tofbb=2*r2/c;

if(tofaa <= tofbb)
 k=int (tofaa/(0.000001));
else
 k=int (tofbb/(0.000001));

kfinal=k+500;
k-=300;
m=0;
do
{
 t=k*0.000001;
 timet[m]=t;
 aa[m]=f1(t,tofaa,theta,theta,r1,r1);
 ba[m]=f1(t,tofba,theta1,theta2,ra,rb);
 bb[m]=f1(t,tofbb,theta,theta,r2,r2);
 ab[m]=f1(t,tofab,theta2,theta1,rb,ra);
 m++;
 k++;
}while(k <= kfinal);

}

// To produce echo signal which is reflected from a corner at range r
// and inclination angle theta:

void  corner(double r, double theta)
{

double thetaa,thetab,tofaa,tofab,tofba,tofbb,t,ra,rb,th;
int k,kfinal,m;

Amax=1.070032;
th=theta*M_PI/180.0;
ra=sqrt(r*r+d*d/4.0-r*d*sin(th));
rb=sqrt(r*r+d*d/4.0+r*d*sin(th));
thetaa=180.0*acos(r*cos(th)/ra)/M_PI;
```



```
thetab=180.0*acos(r*cos(th)/rb)/M_PI;

tofaa=2*ra/c;
tofab=tofba=2*r/c;
tofbb=2*rb/c;

if(tofaa <= tofbb)
 k=int (tofaa/(0.000001));
else
 k=int (tofbb/(0.000001));

kfinal=k+500;
k-=300;
m=0;

do
{
 t=k*0.000001;
 timet[m]=t;
 aa[m]=f1(t,tofaa,thetaa,thetaa,ra,ra);
 ba[m]=f1(t,tofba,theta,theta,r,r);
 ab[m]=f1(t,tofab,theta,theta,r,r);
 bb[m]=f1(t,tofbb,thetab,thetab,rb,rb);
 m++;
 k++;
}while(k <= kfinal);

}

// To produce the echo signals when the sensor pair scans the simulated rc

double signal(double xs,double ys,double phi)
{
/*
   (xs,ys) : coordinates of the sensor pair;
   phi     : angle between horizontal axis and sensor pair whose
             range -180 to 180; */

double r0,thetap,xt,yt,t,flag;
int k,m;
```

```
xt=5.0; /* horizontal length of the room */
yt=5.0; /* vertical length of the room */
flag=0.0;

if(-13.0 < phi && phi < 13.0 )
{
//Target is a plane.
 xt/=2.0;
 yt=0.0;
 r0=xt-xs;
 thetap=phi;
 plane(r0,thetap);
 flag=1.0;

}
else if(30.0 < phi && phi < 60.0)
{
//Target is a corner.
 xt/=2.0;
 yt/=2.0;
 r0=sqrt((xt-xs)*(xt-xs)+(yt-ys)*(yt-ys));
 thetap=phi-180.0*atan2((yt-ys),(xt-xs))/M_PI;
 corner(r0,thetap);
 flag=1.0;

}
else if(77.0 < phi && phi < 103.0)
{
//Target is a plane.
 xt=0.0;
 yt/=2.0;
 r0=yt-ys;
 thetap=phi-90.0;
 plane(r0,thetap);
 flag=1.0;

}
else if(120.0 < phi && phi < 150.0)
{
```



```
 thetap=phi+90.0;
 plane(r0,thetap);
 flag=1.0;

}
else if ( -60.0 < phi && phi < -30.0 )
{
//Target is a corner.
 xt/=2.0;
 yt/=-2.0;
 r0=sqrt((xt-xs)*(xt-xs)+(yt-ys)*(yt-ys));
 thetap=phi-180.0*atan2((yt-ys),(xt-xs))/M_PI;
 corner(r0,thetap);
 flag=1.0;

}

else
{
flag=0.0;
}
return(flag);
}

// To get the maximum amplitude and the TOF of the echo signal:

double getdata(void)
{

double trh,flag;
int k;

flag=1.0;
trh=5*stdn;
vmaxaaa=0.0;
vmaxab=0.0;
vmaxba=0.0;
vmaxbb=0.0;

// To add the Gaussian noise with mean 0 and standard deviation stdn
```



```
//Target is a corner.
 xt/=-2.0;
 yt/=2.0;
 r0=sqrt((xt-xs)*(xt-xs)+(yt-ys)*(yt-ys));
 thetap=phi-180.0*atan2((yt-ys),(xt-xs))/M_PI;
 corner(r0,thetap);
 flag=1.0;

}
else if( 167.0 <phi && phi <= 180.0 || -180.0 <= phi && phi < -167)
{
//Target is a plane.
 xt/=-2.0;
 yt=0.0;
 r0=fabs(xt-xs);
 if( phi > 0.0 )
  thetap=phi-180;
 else
  thetap=phi+180;
 plane(r0,thetap);
 flag=1.0;

}
else if(-150.0 < phi && phi < -120.0)
{
//Target is a corner.
 xt/=-2.0;
 yt/=-2.0;
 r0=sqrt((xt-xs)*(xt-xs)+(yt-ys)*(yt-ys));
 thetap=phi-180.0*atan2((yt-ys),(xt-xs))/M_PI;
 corner(r0,thetap);
 flag=1.0;

}
else if( -103.0 < phi && phi < -77.0)
{
//Target is a plane.
 xt=0.0;
 yt/=-2.0;
 r0=fabs(yt-ys);
```



```
// on the echo waveforms:

for(k=1;k < (M1-2);k++)
{
 ab[k]+=N(0.0,stdn);
 bb[k]+=N(0.0,stdn);
 aa[k]+=N(0.0,stdn);
 ba[k]+=N(0.0,stdn);
}
// To find the maximum amplitude of echo waveforms:

for(k=1;k < (M1-2);k++)
{
 if (fabs(ab[k]) > vmaxab)
 vmaxab=fabs(ab[k]);
 if (fabs(ba[k]) > vmaxba)
 vmaxba=fabs(ba[k]);
 if (fabs(bb[k]) > vmaxbb)
 vmaxbb=fabs(bb[k]);
 if (fabs(aa[k]) > vmaxaaa)
 {
 vmaxaaa=fabs(aa[k]);
 }

}

// To find the TOF of echo waveforms:

k=0;
do
 k++;
while(aa[k] < trh && k<=M1-2);
vtofaa=timet[k];
if(k==M1-2 || vmaxaaa==0.0)
flag=0.0;

k=0;
do
 k++;
while(ba[k] < trh && k<=M1-2);
```

```
vtofba=timet[k];
if(k==M1-2||vmaxba==0.0)
flag=0.0;

k=0;
do
 k++;
while(ab[k] < trh && k<=M1-2);
vtofab=timet[k];
if(k==M1-2||vmaxab==0.0)
flag=0.0;

k=0;
do
 k++;
while(bb[k] < trh  && k<=M1-2);
vtofbb=timet[k];
if(k==M1-2 || vmaxbb==0.0)
flag=0.0;

return(flag);
}

// To find the scaling factor in the assigned belief value to a plane:

double findpmax(double r)
{
plane(r,0.0);
getdata();
pmax=(vmaxaaa-vmaxab);
return(pmax);
}

// To find the scaling factor in the assigned belief value to a corner:

double findcmax(double r)
{
double angle,diff;

angle=-15.0;
```



```
cmax=0.0;

do
{
 corner(r,angle);
 getdata();
 diff=fabs(vmaxab-vmaxaaa)+fabs(vmaxab-vmaxbb);
 if(diff > cmax)
  cmax=diff;
 angle++;
}while(angle <= 15.0);

return(cmax);
}

// Classification of the target primitives:

void findtarget(double xsp, double ysp, double phis)
{
/*
   (xsp,ysp) : coordinates of the sensor pair;
   phip      : angle between horizontal axis and sensor pair whose
               range -180 to 180; */

double raa,rab,rbb;

fplane = fcorner = funknown = 0.0;

// Plane-corner differentiation:

if(maxbb > 5*stdn)
{
if(maxaaa > 5*stdn)
{
if(maxab > 5*stdn)
{

if((maxaaa-maxab) > stdn && (maxbb-maxab) > stdn)
fplane=1.0;
else if((maxab-maxaaa) > stdn || (maxab-maxbb) > stdn)
```



```
fcorner=1.0;
else
funknown = 1.0;

raa=ntofaa*c/2.0;
rab=ntofab*c;
rbb=ntofbb*c/2.0;

// Belief assignment to the plane and estimation of position of this plane:

if(fplane==1.0)
{
 rf=(raa+rbb)/2.0;
 thetaf=180.0*asin((rbb-raa)/d)/M_PI;
 findpmax(rf);
 bap=(maxaaa-maxab)+(maxbb-maxab)/(2*pmax);
 bac=0.0;
 bau=1.0-bap;
 br=(rmax-rf)/(rmax-rmin);
 bth=(thetamax-fabs(thetaf))/thetamax;
 if( -45.0 <= phis && phis < 45.0)
 {
  xtarget=xsp+rf;
  ytarget=ysp;
 }
 else if( 45.0 <= phis && phis <135)
 {
  xtarget=xsp;
  ytarget=ysp+rf;
 }
 else if( 135.0 <= phis && phis <180.0 || -180.0 <= phis && phis <-135.0)
 {
  xtarget=xsp-rf;
  ytarget=ysp;
 }
 else if(-135.0 <= phis  && phis < -45.0)
 {
  xtarget=xsp;
  ytarget=ysp-rf;
 }
```



```
rtr=sqrt(xtarget*xtarget+ytarget*ytarget);
thetat=180*atan2(ytarget,xtarget)/M_PI;
}

// Belief assignment to the corner and estimation of position of this
// corner:

if (fcorner==1.0)
{
 rf=rab/2.0;
 thetaf=180.0*asin((rbb*rbb-raa*raa)/(2.0*rf*d))/M_PI;
 findcmax(rf);
 bac=(fabs(maxab-maxaaa)+fabs(maxab-maxbb))/cmax;
 bap=0.0;
 bau=1.0-bac;
 br=(rmax-rf)/(rmax-rmin);
 bth=(thetamax-fabs(thetaf))/thetamax;
 if( 0.0 <= phis && phis < 90.0)
 {
  xtarget=xsp+rf*cos(M_PI*(phis-thetaf)/180.0);
  ytarget=ysp+rf*sin(M_PI*(phis-thetaf)/180.0);
 }
 else if( 90.0 <= phis && phis <= 180.0)
 {
  xtarget=xsp-rf*cos(M_PI*(180-(phis-thetaf))/180.0);
  ytarget=ysp+rf*sin(M_PI*(180-(phis-thetaf))/180.0);
 }
 else if( -180 <= phis && phis < -90.0)
 {
  xtarget=xsp-rf*cos(M_PI*(180.0-(thetaf-phis))/180.0);
  ytarget=ysp-rf*sin(M_PI*(180.0-(thetaf-phis))/180.0);
 }
 else if( -90.0 <= phis && phis <0.0)
 {
  xtarget=xsp+rf*cos(M_PI*(thetaf-phis)/180.0);
  ytarget=ysp-rf*sin(M_PI*(thetaf-phis)/180.0);
 }
rtr=sqrt(xtarget*xtarget+ytarget*ytarget);
thetat=180*atan2(ytarget,xtarget)/M_PI;
}
```

```
// Belief assignment to an unknown target:

if(funknown==1.0)
{
bap=bac=xtarget=ytarget=rf=thetaf=0.0;
br=0.0;
bth=0.0;
bau=1.0;
rtr=0.0;
thetat=0.0;
}

}
else
{
bap=bac=xtarget=ytarget=rf=thetaf=0.0;
br=0.0;
bth=0.0;
bau=1.0;
rtr=0.0;
thetat=0.0;
}

}
else
{
bap=bac=xtarget=ytarget=rf=thetaf=0.0;
br=0.0;
bth=0.0;
bau=1.0;
rtr=0.0;
thetat=0.0;
}

}
else
{
bap=bac=xtarget=ytarget=rf=thetaf=0.0;
br=0.0;
```



```
bth=0.0;
bau=1.0;
rtr=0.0;
thetat=0.0;
}


}

// To obtain the probability of correct classification,
// misclassification and unknown:

void getstatistics(double phi)
{

if(-13.0 < phi && phi < 13.0 || 77.0 < phi && phi < 103.0 || 167.0 <phi
    && phi <= 180.0 || -180.0 <= phi && phi < -167 || -103.0 < phi
    && phi < -77.0)
{
if(bap >bac )
{
countt+=bap;
count+=bap;
}
else if(bac>bap)
{
countf+=bac;
count+=bac;
}
else
{
countu+=bau;
count+=bau;
}
}
else if(-150.0 < phi && phi < -120.0 || -60.0 < phi && phi < -30.0 ||
        120.0 < phi && phi < 150.0 || 30.0 < phi && phi < 60.0 )
{
if(bap >bac)
{
```



```
countf+=bap;
count+=bap;
}
else if(bac>bap)
{
countt+=bac;
count+=bac;
}
else
{
countu+=bau;
count+=bau;
}
}
else
{

if(bap >bac)
{
countf+=bap;
count+=bap;
}
else if(bac>bap)
{
countf+=bac;
count+=bac;
}
else
{
countu+=bau;
count+=bau;
}

}
}
main()
{

double phi,control,control1,xpos[6],ypos[6],rt[4],phit[4],bp[4],bc[4],
       bu[4],cphik,brk[4],btheta[4],r,theta,conflict,rs,t11,t12,t21,t22,
```



```
       k11,k12,k21,k22,c1,c2,x[6],y[6],determinant,aapp,bapp;
int m,k,l,nsensor,sensor,i,loop;

FILE *fout1;
FILE *fout2;
fout1=fopen("fusion.dat","w");
fout2=fopen("rangefusion.dat","w");

// Number of sensor pairs:
nsensor=sensor=3;
// Number of loop:
loop=5;

// Position of sensor pairs:

xpos[1]=-2.0;
ypos[1]=2.0;
xpos[2]=2.0;
ypos[2]=2.0;
xpos[3]=-0.5;
ypos[3]=1.0;

count=0.0;
countt=0.0;
countf=0.0;
countu=0.0;

stdn=0.0;

if (!Random_Seed)
                set_seed();
        else
                srand(Random_Seed);

do
{

 // sensor pairs scan the simulated room five times:
 for(l=1;l<=loop;l++)
{
```

```
phi=-180.0;

do
{
  sensor=nsensor;
 // Three sensor pairs assigns their belief values to the target:
 for(k=1;k<=3;k++)
 {
  control=signal(xpos[k],ypos[k],phi);
  if( control !=0.0)
  {
   control1=getdata();
   if( control1 !=0.0)
   {
    maxaaa=vmaxaaa;
    maxbb=vmaxbb;
    maxab=vmaxab;
    maxba=vmaxba;
    ntofaa=vtofaa;
    ntofab=vtofab;
    ntofba=vtofba;
    ntofbb=vtofbb;
    findtarget(xpos[k],ypos[k],phi);
    // they find the projected range value of target in a common
    // coordinate frame whose origin is the center of the simulated room:
    rt[k]=rtr;
    phit[k]=thetat;
    rs=xpos[k]*xpos[k]+ypos[k]*ypos[k];
    // rf is the position of target in their coordinate frame.
    if(rf != 0.0 && rt[k] !=0.0)
     cphik=(rt[k]*rt[k]+rf*rf-rs)/(2.0*rt[k]*rf);
    else
     cphik=1.0;

      brk[k]=br*cphik;
      btheta[k]=bth*cphik;

    bp[k]=bap;
    bc[k]=bac;
    bu[k]=bau;
```



```
  }
 }
}

 control=control1=1.0;
 for(k=1;k<=3;k++)
 {
  control*=signal(xpos[k],ypos[k],phi);
  if( control !=0.0)
  {
   control1*=getdata();
  }
 }
if( control !=0.0)
{
if( control1 !=0.0)
{
 if(phi==90.0 && l==1)
 {
  if((brk[1]+brk[2]+brk[3])!=0.0)
  {
  for(i=1;i<=nsensor;i++)
  {
    x[i]=rt[i]*cos(phit[i]*M_PI/180.0);
    y[i]=rt[i]*sin(phit[i]*M_PI/180.0);
    if(rt[i]==0.0)
    sensor--;
  }
  t11=t12=c1=c2=0.0;
  for(i=1;i<=nsensor;i++)
  {
   t11+=x[i]*x[i];
   t12+=x[i];
   c1+=y[i]*x[i];
   c2+=y[i];
  }
  t22=double (sensor);
  t21=t12;
  determinant=t11*t22-t21*t12;
```



```
  k11=t22/determinant;
  k12=-t12/determinant;
  k21=-t21/determinant;
  k22=t11/determinant;
  aapp=k11*c1+k12*c2;
  bapp=k21*c1+k22*c2;
  r=fabs(bapp)/sqrt(aapp*aapp+1.0);
  theta=180.0*atan2(bapp/(aapp*aapp+1.0),-aapp*bapp/(aapp*aapp+1.0))/M_PI;
 }
else
 {
 r=rmax;
 theta=15.0;
 }
 }
if(phi==135.0 && l==1)
{
// To fuse range and azimuth estimates for corner:
if((brk[1]+brk[2]+brk[3])!=0.0)
 r=(brk[1]*rt[1]+brk[2]*rt[2]+brk[3]*rt[3])/(brk[1]+brk[2]+brk[3]);
else
 r=rmax;

if((btheta[1]+btheta[2]+btheta[3])!=0.0)
 theta=(phit[1]*btheta[1]+phit[2]*btheta[2]+phit[3]*btheta[3])
       /(btheta[1]+btheta[2]+btheta[3]);
else
 theta=15.0;

}
// To write the fused range and azimuth estimates to an output file:
if((phi==90||phi==135)&&l=1)
fprintf(fout2,"\n %f %f %f %f %f %f %f %f %f %f",stdn,phi,rt[1],
                    phit[1],rt[2],phit[2],rt[3],phit[3],r,theta);

// To assign belief values to the fused the range and azimuth estimates:
br=(rmax-r)/(rmax-rmin);
bth=(thetamax-fmod(fabs(theta),15.0))/thetamax;
 // To fuse the belief values which are assigned to target by each sensor
 // pair:
```



```
 for(k=2;k<=3;k++)
 {
  conflict=1-bc[k-1]*bp[k]-bc[k]*bp[k-1];
  bp[k]=(bp[k-1]*bp[k]+bp[k]*bu[k-1]+bp[k-1]*bu[k])/conflict;
  bc[k]=(bc[k-1]*bc[k]+bc[k]*bu[k-1]+bc[k-1]*bu[k])/conflict;
  bu[k]=bu[k-1]*bu[k]/conflict;
 }

 bap=bp[3];
 bac=bc[3];
 bau=bu[3];

 getstatistics(phi);
 }
}
phi++;

}while(phi < 180.0);
}

probt=(countt/count);
probf=(countf/count);
probu=(countu/count);

 // To print the probabilities to an output file in the form of: amplitude
 // noise standard deviation (stdn), probability of correct classification
 // (probt), probability of misclassification (probf), probability of
 // unknown (probu).

fprintf(fout1," %.15f %.15f %.15f %.15f \n",stdn,probt,probf,probu);

countt=0.0;
countf=0.0;
countu=0.0;
count=0.0;

stdn+=0.001;
}while(stdn <= 0.02);

}
```

Program 9: A C++ program to assign belief values to plane, corner and acute corner for a single sensor pair:

```
#include <stdio.h>
#include <math.h>
#include <iostream.h>
#include <Complex.h>
#include <stdlib.h>
#include <stddef.h>
#include <string.h>

Complex z[3];
double c=343.3;
double d=0.5;
double rmax=8.0;
double rmin=0.1;
double theta0=12.0;

int N;
double Aaa,Aab,Aba,Abb,taa,tab,tba,tbb,raa,rab,rba,rbb,r,theta,thetac,bp,
       bc,bac,bu,br,btheta,maxac,maxp,maxc,bel[10][10];
char tartype[1000][12];

// For range estimation for an acute corner:

void estimation(void)
{

double c2,c3,c0,c1,c4,rb;
Complex b;
int l;

// c0,c1 and c2 are the coefficients of the second-order polynomial
// in range estimation for an acute corner.

c0=d*d*rab*rab*pow((raa*raa/(rbb*rbb)+1.0),2.0);
c1=(raa*raa/(rbb*rbb)+1.0)*(raa*raa-(1.0/(rbb*rbb))
   *((rab*rab+d*d)*(raa*raa+rbb*rbb)-(rab*rab-d*d)*(rab*rab-d*d)));
c2=pow((raa*raa/(rbb*rbb)-1.0),2.0);
```



```
// z[1] and z[2] are the roots of this polynomial.

z[1]=(-Complex(c1,0.0)+sqrt(Complex(c1,0.0)*Complex(c1,0.0)
     -Complex(4.0,0.0)*Complex(c2,0.0)*Complex(c0,0.0)))
     /(Complex(2.0,0.0)*Complex(c2,0.0000001));
z[2]=(-Complex(c1,0.0)-sqrt(Complex(c1,0.0)*Complex(c1,0.0)
      -Complex(4.0,0.0)*Complex(c2,0.0)*Complex(c0,0.0)))
      /(Complex(2.0,0.0)*Complex(c2,0.0000001));

// To estimate range value r for theta is not equal 0:

for(l=1;l<=2;l++)
 {
   z[l]-=Complex(d*d/2.0,0.0);
   z[l]/=Complex(2.0,0.0);
   z[l]=pow(z[l],0.5);

   if(fabs(imag(z[l]))<=0.004)
    {
      if(real(z[l])> 0.0 && real(z[l]) <=7.0)
       {
         r=real(z[l]);
       }
    }
 }

// To estimate range value r for theta=0:

if(fabs(imag(z[1])) > 0.004 && fabs(imag(z[2])) > 0.004 )
 {
   r=sqrt(d*d*(d*d+rab*rbb-rab*rab-rbb*rbb)/(4.0*((rab-rbb)*(rab-rbb)-d*d)
 }

// To estimate theta and thetac:

rb=(2.0*r*r+d*d/2.0)/((raa*raa/(rbb*rbb)+1.0));
c4=(r/d+d/(4.0*r))*(1.0-raa*raa/(rbb*rbb))/(raa*raa/(rbb*rbb)+1.0);

if(c4>=-1.0 && c4<=1.0)
 {
```



```
   theta=180.0*asin(c4)/M_PI;
   c3=sqrt(rbb*rbb*(raa*raa/(rbb*rbb)+1.0)/(4.0*(2.0*r*r+d*d/2.0)));
   if(c3>=-1.0 && c3<=1.0)
    {
      thetac=180.0*asin(c3)/M_PI;
    }
 }

}

main()
{
FILE *fin;
FILE *fout;

fin=fopen("data.dat","r");
// input file data.dat contains Aaa(theta), Aab(theta), Aba(theta), Abb(theta).
// taa(theta), tab(theta), tba(theta), tbb(theta) in order.
fout=fopen("belief.dat","w");

int k;
double a,b;

maxp=1.0;
maxc=1.0;
maxac=1.0;

k=1;

while(!feof(fin))
 {
   fscanf(fin,"%lf",&b);
   Aaa=b;
   fscanf(fin,"%lf",&b);
   Aab=b;
   fscanf(fin,"%lf",&b);
   Aba=b;
   fscanf(fin,"%lf",&b);
   Abb=b;
   fscanf(fin,"%lf",&b);
```



```
   taa=b;
   fscanf(fin,"%lf",&b);
   tab=b;
   fscanf(fin,"%lf",&b);
   tba=b;
   fscanf(fin,"%lf",&b);
   tbb=b;

raa=c*taa;
rab=c*tab;
rba=c*tba;
rbb=c*tbb;

bp=bc=bac=bu=0.0;

// Acute corner identification:

if( (taa-tab)>0.0 && (tbb-tba)>0.0 )
 {
  bac=(taa-tab)*(tbb-tba);
  bp=0.0;
  bc=0.0;
  estimation();
  br=(rmax-r)/(rmax-rmin);
  btheta=(theta0-fabs(theta))/theta0;
  tartype[k]="ACUTECORNER";
  if(bac>maxac)
   {
     maxac=bac;
   }
 }

// Plane-corner identification:

else if( (tab-taa)>0.0 || (tba-tbb)>0.0 )
 {
  if( (Aaa-Aab)>0.0 && (Abb-Aba)>0.0 )
   {
    bp=(Aaa-Aab)*(Abb-Aba);
    bc=0.0;
```

```
   bac=0.0;
   r=(raa+rbb)/2.0;
   theta=180*asin((raa+rbb)/d)/M_PI;
   br=(rmax-r)/(rmax-rmin);
   btheta=(theta0-fabs(theta))/theta0;
   thetac=0.0;
   tartype[k]="PLANE      ";
   if(bp>maxp)
    {
     maxp=bp;
    }
  }
 else if( (Aab-Aaa)>0.0 || (Aba-Abb)>0.0 )
  {
   bc=fabs((Aab-Aaa))+fabs((Aba-Abb));
   bp=0.0;
   bac=0.0;
   r=(raa*raa+rbb*rbb-d*d/2.0)/2.0;
   theta=180*asin((raa*raa+rbb*rbb)/(2.0*d*r))/M_PI;
   br=(rmax-r)/(rmax-rmin);
   btheta=(theta0-fabs(theta))/theta0;
   thetac=0.0;
   tartype[k]="CORNER     ";
   if(bc>maxc)
    {
     maxc=bc;
    }
  }
 else
  {
   bp=0.0;
   bc=0.0;
   bac=0.0;
   r=0.0;
   theta=0.0;
   thetac=0.0;
   br=0.0;
   btheta=0.0;
   tartype[k]="UNKNOWN    ";
  }
```



```
 }
else
 {
  bp=0.0;
  bc=0.0;
  bac=0.0;
  r=0.0;
  theta=0.0;
  thetac=0.0;
  br=0.0;
  btheta=0.0;
  tartype[k]="UNKNOWN    ";
 }

bu=1.0-(bp+bc+bac);

bel[k][1]=bp;
bel[k][2]=bc;
bel[k][3]=bac;
bel[k][4]=bu;
bel[k][5]=r;
bel[k][6]=theta;
bel[k][7]=thetac;
bel[k][8]=br;
bel[k][9]=btheta;

k++;
}

N=k-1;

for(k=1;k<N;k++)
{
// Scaling of belief values:

a=bel[k][1]/maxp;
bel[k][1]=a;
a=bel[k][2]/maxc;
bel[k][2]=a;
a=bel[k][3]/maxac;
```



```
bel[k][3]=a;
bel[k][4]=1.0-(bel[k][1]+bel[k][2]+bel[k][3]);

// To print the belief values and estimated parameters: b(plane), b(corner),
// b(acute corner), b(unknown), r, theta, thetac which is assigned to 0 for
// plane and corner, b(r), b(theta) in order:

fprintf(fout,"\n %s %.15f %.15f %.15f %.15f %.15f %.15f %.15f %.15f %.15f",
        tartype[k],bel[k][1],bel[k][2],bel[k][3],bel[k][4],bel[k][5],
        bel[k][6],bel[k][7],bel[k][8],bel[k][9]);
}

fclose(fin);
fclose(fout);

}
```

Program 10: A C++ program for feature fusion for three sensor pairs where the features are: plane, corner and acute corner:

```
#include <stdio.h>
#include <math.h>
#include <iostream.h>
#include <stdlib.h>
#include <stddef.h>
#include <string.h>

double rmax=8.0;
double rmin=0.1;
double theta0=12.0;

int N;

double bp,bc,bac,bu,bps[4],bcs[4],bacs[4],bus[4];

main()
{
FILE *fin1;
FILE *fin2;
FILE *fin3;
```



```
FILE *fout;

fin1=fopen("data1.dat","r");
fin2=fopen("data2.dat","r");
fin3=fopen("data3.dat","r");
fout=fopen("fusedbeliefs.dat","w");

int k;
double b;

k=1;

while(!feof(fin1) && !feof(fin2) && !feof(fin3))
{
 // To read the belief values which are assigned to target by each sensor
 // from the corresponding input files:
fscanf(fin1,"%lf",&b);
bps[1]=b;
fscanf(fin1,"%lf",&b);
bcs[1]=b;
fscanf(fin1,"%lf",&b);
bacs[1]=b;
fscanf(fin1,"%lf",&b);
bus[1]=b;
fscanf(fin2,"%lf",&b);
bps[2]=b;
fscanf(fin2,"%lf",&b);
bcs[2]=b;
fscanf(fin2,"%lf",&b);
bacs[2]=b;
fscanf(fin2,"%lf",&b);
bus[2]=b;
fscanf(fin3,"%lf",&b);
bps[3]=b;
fscanf(fin3,"%lf",&b);
bcs[3]=b;
fscanf(fin3,"%lf",&b);
bacs[3]=b;
fscanf(fin3,"%lf",&b);
bus[3]=b;
```

```
 // To fuse the belief values which are assigned to target by each sensor pair
  for(k=2;k<=3;k++)
  {
   conflict=1-bcs[k-1]*bps[k]-bcs[k]*bps[k-1]-bcs[k]*bacs[k-1]
            -bacs[k]*bcs[k-1]-bacs[k-1]*bps[k]-bps[k-1]*bacs[k];
   bps[k]=(bps[k-1]*bps[k]+bps[k]*bus[k-1]+bps[k-1]*bus[k])/conflict;
   bcs[k]=(bcs[k-1]*bcs[k]+bcs[k]*bus[k-1]+bcs[k-1]*bus[k])/conflict;
   bcs[k]=(bacs[k-1]*bacs[k]+bacs[k]*bus[k-1]+bacs[k-1]*bus[k])/conflict;
   bus[k]=bus[k-1]*bus[k]/conflict;
  }

  bp=bps[3];
  bc=bcs[3];
  bac=bacs[3];
  bu=bus[3];

// To write the fused belief values to an output file in the form of b(plane),
// b(corner), b(acute corner), b(unknown).
fprintf(fout1," %.15f %.15f %.15f %.15f \n",bp,bc,bac,bu);

 }

}
```

Program 11: A PlaNet program to calculate the weight values in all layers of neural networks using backpropagation algorithm:

```
# Definitions of layers:

layer Input 155   # number of neurons in the input layer
layer Hidden1 50  # number of neurons in the first hidden layer
layer Hidden2 25  # number of neurons in the second hidden layer
layer Output 7    # number of neurons in the output layer

# definitions of input/target buffers:

target 7
input 155
```



```
# Definitions of connections:

connect InputHidden1 Input to Hidden1
connect Hidden1Hidden2 Hidden1 to Hidden2
connect Hidden2Output Hidden2 to Output

# procedure for activating network forward:

procedure activate
input Input
forward InputHidden1
activation Hidden1
forward Hidden1Hidden2
activation Hidden2
forward Hidden2Output
activation Output
target Output
end

# procedure for training network:

procedure learn
call activate
backward Hidden2Output
delta Hidden2
backward Hidden1Hidden2
delta Hidden1
learn InputHidden1
learn Hidden1Hidden2
learn Hidden2Output
learnbias Output
learnbias Hidden2
learnbias Hidden1
end
```

Program 12: A Matlab program to compute the output of the neural networks. Program 10 can also be used for this purpose in PlaNet environment but it is not convenient for testing the network with many testing patterns.

```
function o=nnetwork(x)
```



```
% input matrix x contains the training patterns the number of column is
% the number of inputs to the neural network and the number of rows is
% the number of patterns to be tested.

% To load the weight matrices of neural network :

load W1.dat;
load W2.dat;
load W3.dat;

x=x';
o=[];
[k,l]=size(x);
d=1.0*ones(size([1:l]));
d1=[x;d];
x=d1;
x1=W1*x;
v1=activation(x1); % finds the input layer response
[k,l]=size(v1);
d=1.0*ones(size([1:l]));
d1=[v1;d];
v1=d1;
x2=W2*v1; % finds the hidden layer response
v2=activation(x2);
[k,l]=size(v2);
d=1.0*ones(size([1:l]));
d1=[v2;d];
v2=d1;
x3=Wtotal3*v2;
v3=activation(x3); % finds the output
[k,l]=size(v3);
for j=1:l
o1=v3(:,j);
[a,b]=max(o1);
o1=zeros(size(o1));
o1(b)=1.0;
o=[o;o1'];
end;
```


Program 13: A matlab program which is a sigmoid function used as an activation function in Program 12:

```
function y=activation(x);
[k,l]=size(x);
for i=1:k
for j=1:l
y(i,j)=1.0/(1.0+exp(-x(i,j)));
end;
end;
```

REFERENCES

[1] S. S. Blackman and T. J. Broida, "Multiple sensor data association and fusion in aerospace applications," *Journal of Robotic Systems*, vol. 7, pp. 445–485, June 1990.

[2] A. N. Steinberg, "Threat management system for combat aircraft," *Proceedings of the 1987 Tri-Service Data Fusion Symposium*, 1:532–554, Naval Air Development Center, Warminster, PA, June 1987.

[3] R. C. Luo and M. H. Lin, "Robot multi-sensor fusion and integration: Optimum estimation of fused sensor data," *Proceedings of 1988 IEEE International Conference on Robots and Automation, Philadelphia,* PA, pp. 24–29, April 1988.

[4] R. Kuc and B. V. Viard, "A physically-based navigation strategy for sonar-guided vehicles," *The International Journal of Robotics Research*, vol. 10, pp. 75–87, April 1991.

[5] R. Kuc and M. W. Siegel, "Physically-based simulation model for acoustic sensor robot navigation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-9, pp. 766–778, November 1987.

[6] J. L. Crowley, "Navigation for an intelligent mobile robot," *IEEE Transactions on Robotics and Automation*, vol. RA-1, pp. 31–41, March 1985.

[7] J. J. Leonard and H. F. Durrant-Whyte, "Mobile robot localization by tracking geometric beacons," *IEEE Transactions on Robotics and Automation*, vol. 7, pp. 376–382, 1991.

[8] R. Kuc, "Three-dimensional tracking using qualitative bionic sonar," *Robotics and Autonomous Systems*, vol. 11, pp. 213–219, 1993.

[9] J. Borenstein and Y. Koren, "Obstacle avoidance with ultrasonic sensors," *IEEE Transactions on Robotics and Automation*, vol. RA-4, pp. 213–218, April 1988.

[10] L. Kleeman and R. Kuc, "Mobile robot sonar for target localization and classification," *International Journal of Robotics Research*, vol. 14, pp. 295–318, August 1995.

[11] C. Biber, S. Ellin, E. Sheck and J. Stempeck, "The Polaroid ultrasonic ranging system," in *Proceedings 67'th Audio Engineering Society Convention*, 1990. Reprinted in Polaroid Ultrasonic Ranging System Handbook.

[12] Polaroid Corporation. *Ultrasonic Components Group.* 119 Windsor St., Cambridge, MA 02139, 1990.

[13] B. Barshan and R. Kuc, "Differentiating sonar reflections from corners and planes by employing an intelligent sensor," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, pp. 560–569, June 1990.

[14] J. Manyika and H. F. Durrant-Whyte. *Data Fusion and Sensor Management: A Decentralized Information-Theoretic Approach.* Ellis Horwood, New York, 1994.

[15] M. L. Hong and L. Kleeman, "A low sample rate 3-D sonar sensor for mobile robots," in *Proceedings IEEE International Conference on Robotics and Automation*, pp. 3015–3020, Nagoya, Japan, May 21–27, 1995.

[16] L. Kleeman and H. Akbarally, "A sonar sensor for accurate 3-D target localization and classification," in *Proceedings IEEE International Conference on Robotics and Automation*, pp. 3003–3008, Nagoya, Japan, May 21–27, 1995.

[17] M. L. Hong and L. Kleeman, "Analysis of ultrasonic differentiation of three-dimensional corners, edges and planes," in *Proceedings IEEE International Conference on Robotics and Automation*, pp. 580–584, Nice, France, May 12–14, 1992.

[18] A. M. Flynn, "Combining sonar and infrared sensors for mobile robot navigation," *The International Journal of Robotics Research*, vol. 7, pp. 5–14, December 1988.

[19] H. F. Durrant-Whyte, "Sensor models and multisensor integration," *The International Journal of Robotics Research*, vol. 7, pp. 97–113, December 1988.

[20] R. C. Luo and M. G. Kay. *Multisensor Integration and Fusion for Intelligent Machines and Systems.* Ablex Puplishing Corporation, Norwood, NJ, 1995.

[21] S. Watanabe and M. Yoneyama, "An ultrasonic visual sensor for three-dimensional object recognition using neural networks," *IEEE Transactions on Robotics and Automation*, vol. 8, pp. 240–249, 1992.

[22] B. Widrow and R. Winter, "Neural nets for adaptive filtering and adaptive pattern recognition," *IEEE Computer*, vol. 21, pp. 25–40, March 1988.

[23] C. Prieve and D. Marchette, "An application of neural networks to a data fusion problem," *1987 Tri-Service Data Fusion Technical Proceedings, John Hopkins University, Maryland*, pp. 226–236, June 1987.

[24] G. B. Willson, "Radar classification using a neural network," *Optical Engineering and Photonics in Aerospace Sensing: Application of Neural Networks, Proceedings of SPIE*, vol. 1294, pp. 200–210, April 1990.

[25] S. Nazlıbilek, A. Erkmen and I. Erkmen, "SENS-PERCEPTOR: Image-based evidence formation module as a logical sensor for robot hand preshaping," in *Proceedings IEEE International Symposium on Intelligent Control*, pp. 326–331, Chicago, IL, August, 1993.

[26] J. Zemanek, "Beam behaviour within the nearfield of a vibrating piston," *The Journal of the Acoustical Society of America*, vol. 49, pp. 181–191, January 1971.

[27] M. S. Smith. *Introduction to Antennas.* Springer-Verlag, New York, 1988.

[28] B. Barshan, "Location and curvature estimation of spherical targets for robotics applications using a multiple sonar configuration," *Journal of Robotic Systems*, 1996. Submitted for publication.

[29] H. Peremans, K. Audenaert and J. M. Van Campenhout, "A high-resolution sensor based on tri-aural perception," *IEEE Transactions on Robotics and Automation*, vol. 9, pp. 36–48, February 1993.

[30] G. Shafer. *A Mathematical Theory of Evidence.* Princeton:Princeton University Press, 1976.

[31] R. R. Murphy, "Robust sensor fusion for teleoperations," in *Proceedings IEEE International Conference on Robotics and Automation*, pp. 572–577, Atlanta, GA, May 2–6 1993.

[32] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann Publishers, Inc., San Mateo, CA, USA.

[33] J. O. Berger. *Statistical Decision Theory and Bayesian Analysis.* Springer-Verlag, Berlin, Germany, 1988.

[34] D. L. Hall. *Mathematical Techniques in Multisensor Data Fusion.* Artech House, Norwood, MA, 1992.

[35] Ö. I. Bozma. *A Physical Model-Based Approach to Analysis of Environments using Sonar.* PhD thesis, Yale University, New Haven, CT, May 1992.

[36] R. P. Lippman, "An introduction to computing with neural nets," *IEEE ASSP Magazine*, pp. 4–22, April 1987.

[37] P. J. Werbos, "Backpropagation through time: what it does and how to do it," *Proceedings of the IEEE*, vol. 78, pp. 1550–1560, October 1990.

[38] D. H. Ackley, G. E. Hinton and T. J. Sejnowski, "A learning algorithn for boltzmann machine," *Cognitive Sci.*, vol. 9, pp. 147–169, 1985.

[39] R. Hecht-Neilsen, "Counterpropagation networks," *Appl. Opt.*, vol. 26, pp. 4979–4984, December 1987.

[40] B. Windrow and M. A. Lehr, "30 years of adaptive neural networks: Perceptron, madaline, and backpropagation," *Proceedings of the IEEE*, vol. 78, pp. 1415–1442, September 1990.

[41] B. Barshan. *A Sonar-Based Mobile Robot for Bat-Like Prey Capture.* PhD thesis, Yale University, New Haven, CT, December 1991. University of Michigan Microfilms, order number 9224325.

[42] R. Kuc and B. Barshan, "Navigating vehicles through an unstructured environment with sonar," in *Proceedings IEEE International Conference on Robotics and Automation*, pp. 1422–1427, Scottsdale, AZ, May 14–19, 1989.